# A MULTI-OBJECTIVE PROGRAMMING PERSPECTIVE TO STATISTICAL LEARNING PROBLEMS

A Thesis
Presented to
The Academic Faculty

by

Sibel Yaman

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December 2008

# A MULTI-OBJECTIVE PROGRAMMING PERSPECTIVE TO STATISTICAL LEARNING PROBLEMS

Approved by:

Professor Chin-Hui Lee, Advisor
School of Electrical and Computer Engineering
*Georgia Institute of Technology*

Professor James H. McClellan
School of Electrical and Computer Engineering
*Georgia Institute of Technology*

Professor Fred Juang
School of Electrical and Computer Engineering
*Georgia Institute of Technology*

Professor Anthony Yezzi
School of Electrical and Computer Engineering
*Georgia Institute of Technology*

Professor Evans Harrell
Department of Mathematics
*Georgia Institute of Technology*

Date Approved: 3 October 2008

*To my family,*

*Gulgez, Ziya, Emel and Sebnem Yaman,*

*and Can Kizilkale*

# ACKNOWLEDGEMENTS

I would like to express my immense appreciation for my advisor, Professor Chin-Hui Lee. I owe him an enormous debt of gratitude for his excellent guidance, support, and encouragement throughout my graduate studies at Georgia Tech. It has been a privilege to have him as my advisor and the effect of his efforts on my research and graduate experience cannot be overstated. He has spent so much time showing me how to proceed in a research project as well as teaching how to present my work in a scholarly article.

I would like to take this opportunity to thank Professor James H. McClellan for his assistance at my toughest times at Georgia Tech. I was also fortunate to be a graduate teaching assistant of his course. He has set me a role-model as a teacher and has helped me realize what a good impact a professor can make on the lives of his or her students.

I would like to express my sincere gratitude to the Microsoft Research Speech Technologies Group for their generous support during the last two years of my graduate studies. I have enormously benefited the research experience I gained while I was an intern there and I truly appreciate their support and kindness. It has been a privilege to have collaborated with Dr. Li Deng, Dr. Ye-Yi Wang, Dr. Dong Yu, and Dr. Alex Acero.

I also thank Professor Jen-Tzung Chien for his collaborations in the language modeling project. I greatly benefited from his knowledge and expertise on the subject.

My special thanks are due to Professors James H. McClellan and Prof. Fred Juang for serving in the reading committee of my thesis. They have provided me with valuable feedback on my research. I also thank Professors Anthony Yezzi and

Evans Harrell for serving in my thesis committee and for providing me with valuable feedback.

Many thanks go to the professors and the staff members of the Center for Signal and Image Processing (CSIP) and the Georgia Institute of Technology. Thanks to their efforts, the students are being offered all the resources and the environment they need to do their research.

I have also been very fortunate to have the accompany of great friends at Georgia Tech. In particular, I owe special thanks to Sevgi Zubeyde and Ali Cafer Gurbuz for their sincere friendships and for sharing many memorable moments together. I also owe special thanks to Selcuk Uluagac, Antonio Moreno, and Marco Sinischialni for their cheerful talks and for their dependable friendships. It has been a great pleasure having Pinar Korkmaz Ayhan, Sanem Sariel Talay, Volkan Cevher, David Boivin, Nicholas Gestaud, Tushar Kumar, Majid Fozunbal, Nazanin Rahnavard, Jinyu Li, Jeremy Reed, Chengyuan Ma, Yu Tsao, and Nhi Nguyen as friends and it will be as pleasant to remember our memories together. I also thank Maz Kosma for leading and doing much of the work for the many activities for the Turkish community at Georgia Tech. I have also great friends from all stages of my life, and in particular, Ozlem Eskil deserves my special thanks for making herself available whenever I needed.

Finally, I would like to express my deepest gratitude to the very special ones I am blessed with, although I cannot possibly thank them as much as they deserve. I thank my parents Gulgez and Ziya Yaman, and my sisters Emel Yaman Sezer and Sebnem Yaman, for their endless love, support, encouragement, and sacrifice throughout my life and during my Ph.D. studies. I thank Can Kizilkale for his warm love since the time I know him and for his exceptional friendship and accompany. I could write pages about the positive influences they have had on my career, and indeed, on my whole life but I would like to finalize my acknowledgements by stating that this work would not have been possible without their devoted love and faith in me.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# SUMMARY

It has been increasingly recognized that realistic problems often involve the consideration of a tradeoff among many conflicting objectives. Traditional methods aim at satisfying multiple objectives by combining them into a global cost function, which in most cases overlooks the underlying tradeoffs between the conflicting objectives. This raises the issue about how different objectives should be combined to yield a final solution. Moreover, such approaches promise that the chosen overall objective function is optimized over the training samples. However, there is no guarantee on the performance in terms of the individual objectives since they are not considered on an individual basis.

Motivated by these shortcomings of traditional methods, the objective in this dissertation is to investigate theory, algorithms, and applications for problems with competing objectives and to understand the behavior of the proposed algorithms in light of some applications. We develop a multi-objective programming (MOP) framework for finding compromise solutions that are satisfactory for each of multiple competing performance criteria. The fundamental idea for our formulation, which we refer to as iterative constrained optimization (ICO), evolves around improving one objective while allowing the rest to degrade. This is achieved by the optimization of individual objectives with proper constraints on the remaining competing objectives. The constraint bounds are adjusted based on the objective functions obtained in the most recent iteration. An aggregated utility function is used to evaluate the acceptability of *local* changes in competing criteria, i.e., changes from one iteration to the next.

Conflicting objectives arise in different contexts in many problems of speech and language technologies. In this dissertation, we consider two applications. The first application is language model (LM) adaptation, where a general LM is adapted to a specific application domain so that the adapted LM is as close as possible to both the general model and the application domain data. Language modeling and adaptation is used in many speech and language processing applications such as speech recognition, machine translation, part-of-speech tagging, parsing, and information retrieval.

The second application is automatic language identification (LID), where the standard detection performance evaluation measures *false-rejection* (or miss) and *false-acceptance* (or false alarm) rates for a number of languages are to be simultaneously minimized. LID systems might be used as a pre-processing stage for understanding systems and for human listeners, and find applications in, for example, a hotel lobby or an international airport where one might speak to a multi-lingual voice-controlled travel information retrieval system.

This dissertation is expected to provide new insights and techniques for accomplishing significant performance improvement over existing approaches in terms of the individual competing objectives. Meantime, the designer has a better control over what is achieved in terms of the individual objectives. Although many MOP approaches developed so far are formal and extensible to large number of competing objectives, their capabilities are examined only with two or three objectives. This is mainly because practical problems become significantly harder to manage when the number of objectives gets larger. We, however, illustrate the proposed framework with a larger number of objectives.

# CHAPTER I

# INTRODUCTION

It has been increasingly recognized that realistic problems often involve the consideration of a tradeoff among many conflicting goals. Traditional methods aim at satisfying multiple objectives by forming a global objective function and solving the resulting problem through the use of classical single-objective programming (SOP) methods. Combining several competing objectives into an overall objective function, such SOP-based approaches promise that the chosen overall objective function is optimized. However, there is no guarantee on the performance of the individual objectives since they are not considered separately. What makes things worse is that in realistic problems, one or more objectives tend to dominate the optimization process.

It is important to note that once an overall objective function is set, the original problem is simplified to a problem with a definitive tradeoff relation among the objectives. However, the tradeoff relations are often unknown a priori, and thus it is desired to be able to have a high degree of freedom to achieve different levels of tradeoff. Undoubtedly, it would easily become overwhelming for the designer to find an overall objective function that achieves desirable levels for the individual objectives. For these reasons, we articulate that methods of traditional SOP are not enough for realistic problems with competing objectives.

The problems that involve complex tradeoff relationships among many conflicting objectives are the subject of *multi-objective programming* (MOP). MOP attempts to simultaneously optimize a set of conflicting objectives by solving

$$\min_{\mathbf{w}} \{f_1(\mathbf{w};\mathbf{x}), f_2(\mathbf{w};x), ..., f_K(\mathbf{w};x)\}$$

where $\mathbf{w}$ denotes the unknowns and $\mathbf{x}$ denotes the data samples we collect to infer a statistical model. In the rest of our discussion, we will omit $\mathbf{x}$ since the data samples, $x$, are fixed for a given data set. In general, an improvement with regard to one objective, $f_i$, causes a deterioration of another, $f_j$. This corresponds to the situation that the objective functions are *at least partially conflicting*, meaning that they are conflicting at least in some regions of the search space for $\mathbf{w}$.

In this dissertation we aim at investigating MOP-based methods for speech and natural language applications in the pursuit of the following goals:

- each objective function is to attain a satisfactory level,
- a high degree of flexibility is to be present to achieve different levels of tradeoffs,
- an overall objective function is avoided since it is hard to determine a realistic one a priori,
- no single objective function is to dominate the optimization process.

We next describe the most common scenarios where multi-objective problems arise. In Section 1.2, we describe some statistical learning contexts in which multiple objectives arise. In Section 1.3, the two specific applications that are considered in this dissertation, namely, spoken language identification and language model adaptation, are described. In Section 1.4, the conflicting objectives involved in these two applications are presented. Finally in Section 1.5, the contributions and the organization of this dissertation are presented.

## *1.1 Contexts that Give Rise to Multiple Objectives*

Many engineering problems can be formulated as multi-objective problems. This section is devoted to a brief overview of different reasons underlying the need for MOP in engineering problems.

### 1.1.1 Standard Multi-Objective Problems

In some problems, all the objectives are clear, measurable goals that one would genuinely like to optimize. Assuming all important criteria have been included as objectives, one may be unsure about their relative importance but can be certain about how the ideal solution will be. An example of this type of problem setting is the optimization of detection systems where tradeoffs exist between the miss and false-alarm rates.

### 1.1.2 Counter-balance for Bias

There are problems where MOP may be used as a tool to counter-balance a measurement bias affecting an objective function. Such a measurement bias is, for example, encountered in alignment problems, where short alignments can be trivially obtained and the number of mismatches automatically increases with the length of the alignment. Mathematically, this setting can be described as follows, assuming just one (primary) objective to be optimized:

$$f(\mathbf{w}) = f_1(\mathbf{w}) + \alpha \cdot f_2(\mathbf{w}),$$

where $f_1$ is an ideal, unbiased measure of the primary objective, $f_2$ is a bias term and $f$ is the measurable but biased sum of the two. This problem may, instead, be formulated as

$$\min_{\mathbf{w}} \{f_1(\mathbf{w}), f_2(\mathbf{w})\}.$$

### 1.1.3 Multiple Source Integration

MOP may be used to integrate noisy data from multiple sources. Hence, in this setting, it is used as an alternative to an a priori or a posteriori integration technique. The problems where this approach is used are originally single-objective. However,

multiple noisy views of the data need to be integrated, as their combined use may yield better results than the use of data from a single information source. Mathematically, this setting can be described by a set of objective functions:

$$\begin{aligned} f_1(\mathbf{w}) &= \tilde{f}_1(\mathbf{w}) + \eta_1 \\ f_2(\mathbf{w}) &= \tilde{f}_2(\mathbf{w}) + \eta_2 \\ &... \\ f_K(\mathbf{w}) &= \tilde{f}_K(\mathbf{w}) + \eta_K \end{aligned}$$

where the function value of each objective function, $f_k$, is equal to the value of an ideal function, $\tilde{f}_k$, with some unknown random noise, $\eta_k$, on it, for $k = 1, ..., K$. In some cases, the $\tilde{f}_k$'s are all identical, i.e., the 'views' of the data arise from the same type of measurement but taken at different times. By formulating the problem as a multi-objective problem, the impact of the noise may be reduced, if it is reasonably uncorrelated with the solution space.

### 1.1.4 Multi-Objectivization

MOP may also be used solely as a way to obtain improved search 'guidance' in what is essentially a single-objective problem. Assuming a single objective that is measurable, a problem may still be difficult because of its search landscape. A decomposition of the primary objective into several different functions (each function either defined over all of the variables or a subset of them) may help to separate out the conflicting aspects of the problem, thus reducing the number of local optima 'seen' by a search algorithm [52]. In the second case, the use of extra helper objectives in addition to the primary objective may provide helpful guidance [44, 52].

## *1.2 Statistical Learning Tasks that Give Rise to Multiple Objectives*

There have been dramatic improvements in the capability and performance of large-scale data processing systems over the last few decades. This progress may be largely attributed to advances in statistical modeling techniques and procedures for automatic learning from large corpora. *Statistical modeling* addresses the problem of constructing a stochastic model to predict the behavior of a random process. In constructing this model, we typically have at our disposal *sample data* from the process. Given this sample, representing an incomplete state of knowledge about the process, the modeling problem is to parlay this knowledge into a representation of the process. We can then use this representation to make predictions of the future behavior of the process.

The field of speech and natural language technologies is one such area that makes extensive use of advanced statistical methods. Speech and natural language are the most natural means for communicating between humans, and interacting with a computer in the same natural way has been a long-held desire to humans. There are many problems, though, that hinder us from using speech and natural languages for developing such applications. Among the most important problems are the complexity of training the computer to recognize spontaneous speech even in adverse conditions and the difficulty to learn the natural language grammar as good as humans do.

### 1.2.1 Applications with Automatic Speech Recognition

There have been many problems investigating the most natural means of interacting with computers. *Automatic speech recognition* (ASR) investigates the use of computer programs to convert human speech into text, which enables us to talk to a computer. The resulting text can be used in many applications, for instance, to help reduce the need for human operators.

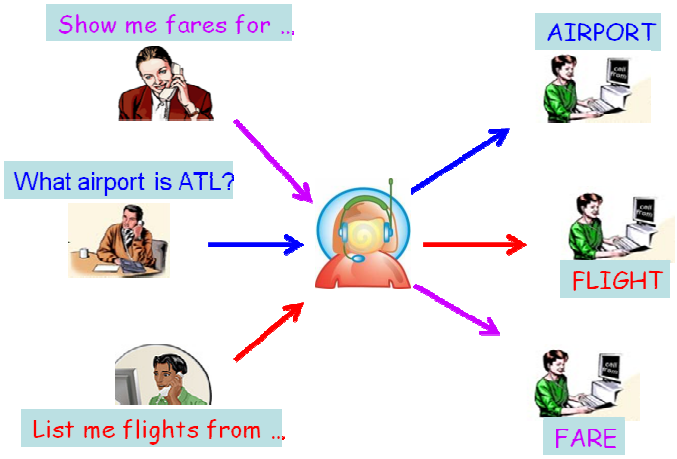



**Figure 1:** An illustration of an SUC system in which an ASR system is in the front-end.

An ASR system can be utilized as a *front-end system*, in which the transcribed text is used to take an action. As an example, consider a *spoken utterance classification* (SUC) task [110], as depicted in Figure 1, where user requests such as

"Show me flights from Boston to Newark."

are processed. The system acts on this utterance by classifying it into application-specific pre-determined departments (i.e., categories) such as FLIGHT, FARE, and AIRPORT. Based on the classification decision, the system transfers the call into the adequate department.

An ASR system can also be utilized as a *back-end system*, in which an ASR engine is preceded by another system. For instance, consider a multi-lingual customer service system, as depicted in Figure 2, where a user calls the system to make a flight or hotel reservation in his/her native language. The multi-lingual system first identifies the user's language and then runs the ASR system specific to his/her language. This system consists of a front-end automatic *spoken language identification* (LID) system and a back-end ASR system. It is the LID system's responsibility to reliably identify the user's language; determining a wrong language would frustrate potential customers.

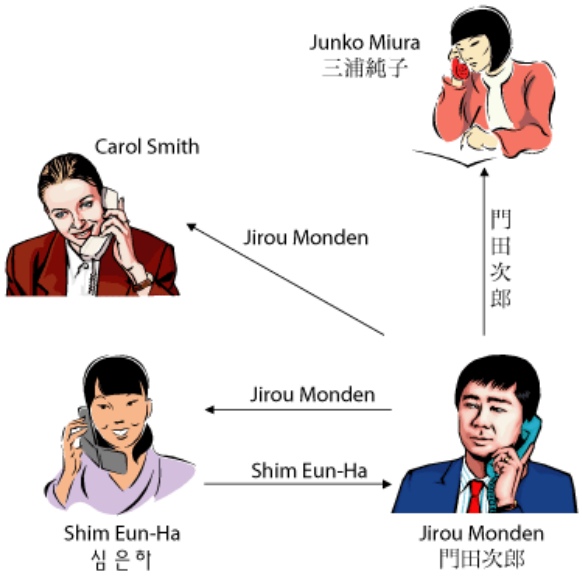



**Figure 2:** An illustration of an LID system in which ASR is in the back-end.

Real-life applications are complex systems that require achieving a number of design objectives. Their successful deployment crucially depend on reasonable performance in terms of all these objectives. For instance, in the LID application, we would like to identify each language correctly, which requires the optimization of two conflicting objectives: (1) to detect the spoken utterances of the language spoken in it, and (2) to not detect the spoken utterances of a language as instances of another language. In detection theory terminology, the former objective is known as detection rate and it is directly related to the *miss rate*, $P(Miss)$, or equivalently the *false-rejection rate*, $E^{FR}$,

$$P(Miss) = E^{FR} = \frac{c_{NEG|POS}}{c_{POS}}, \tag{1.2.1}$$

where $c_{NEG|POS}$ is the number of positive instances recognized as negative instances and $c_{POS}$ is the number of positive instances. The latter objective is known as the *false-alarm rate*, $P(False-alarm)$, or equivalently the *false-acceptance rate*, $E^{FA}$,

$$P(False-alarm) = E^{FA} = \frac{c_{POS|NEG}}{c_{NEG}}, \tag{1.2.2}$$

where $c_{POS|NEG}$ is the number of negative instances recognized as positive instances

and $c_{NEG}$ is the number of negative instances. A desirable LID system could be tolerant to an error rate of up to, for example, 5% for both objectives for each of the languages. It might be possible that the system we design performs well for some languages, say, for KOREAN, but poorly for some other language, say, for HINDI. With such a system, the Korean-speaking customers would not have any problem but Hindi-speaking customers would find it frustrating. The designer may hence want to change his/her system to reduce the frequency of misidentifications of utterances spoken in HINDI by assigning a higher importance to the corresponding error rates. Unfortunately, doing so results in increasing, for example, the frequency of recognizing GERMAN utterances as RUSSIAN. The designer has no way to estimate such complex interactions, especially when statistical techniques are being used to build a model. If he/she could know, it could be possible to formulate an overall design objective that will handle all such error rates in a manner that the designer would be satisfied with the frequencies of all of these unavoidable errors.

### 1.2.2 Language Modeling for ASR

In addition to such explicit design objectives, there is an intrinsic *detail versus reliability* tradeoff (or accuracy versus generalization tradeoff, or specificity versus reliability tradeoff) in every statistical modeling problem. The statistical model should have enough detail that the uncertainty that the model leaves is minimized. It should also be reliable, which requires that the model generalizes well to unseen data. This tradeoff is typically handled with *regularization* (or penalization) methods to avoid over-fitting the data and hence to improve the generalization capabilities [74]. Traditionally, regularization is conduced by including an additional *penalty term* in the cost function of a learning algorithm to penalize too much dependence on the data. In fact, regularization incorporates a *prior information* into the cost function that favors some parameters to others. For example, the penalty term may be in the form of sum

of squared-magnitudes of system parameters, which is equivalent to incorporating a Gaussian prior for the parameters into the cost function.

One of the statistical problems that involve a detail versus reliability tradeoff is language model adaptation. A *language model* (LM) is a means to predict the probability of a word sequence in a given context. For instance, we may want to determine how likely it is to observe a sentence like

"The oil prices will fall down to $100 a barrel."

in the news. LMs find applications in speech recognition, machine translation, part-of-speech tagging, parsing, information retrieval, and optical character recognition. In ASR, an LM is used as a linguistic information source in addition to acoustic information available in speech signal to estimate word sequences that are more likely than others for a given language. For instance, an LM makes it possible to distinguish homonyms (i.e., words with different spelling but with same pronunciation) "to, two, too" in a given speech signal.

Most approaches to handle natural language grammar are classified into two categories: rule-based approaches and statistical approaches. The *rule-based approaches* aim at developing methods to specify the natural language grammar as a set of rules which is accurate, but difficult to acquire or learn automatically. All these rules are chosen carefully by language experts such as linguists, and therefore they need to be done manually. Unlike the rule-based approaches, the *statistical approaches* have the advantage that extensive human knowledge or manual work is not needed. The statistical approaches model the language grammar via a set of parameters that can be learned automatically from a reasonable-size training data.

When the data is insufficient to infer a reliable LM, we take a general model and *adapt* it to the domain that we are interested in. When doing so, the use of the general model ensures that the resulting model will be reliable whereas the use of the limited application-specific data ensures that the resulting model will be specific to

the application that we are interested in. Too much dependence on either of these components will make an impact in favor of either the specificity or the reliability of the model. Therefore, a question to answer in language model adaptation is how much one should depend on these two components.

## *1.3 General Setup for Applications*

Statistical modeling makes it possible to make predictions of the future behavior of the process. The prediction task is called *classification* when we predict qualitative outputs, and *regression* when we predict quantitative outputs. Formally speaking, we assume that there exists a mapping from an input space, $\mathbf{X}$, to an output space, $Y$, where $Y$ is a set of class labels in the case of classification, and a space of real values in the case of regression. In the common *supervised pattern recognition* task, we observe the behavior of the random process for some time, collecting a large number of samples $(\mathbf{x}_i, y_i)$. Typically, a particular pair, $(\mathbf{x}, y)$, will either not occur at all in the sample, or will occur at most a few times. In *unsupervised learning*, one is given a dataset, $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$, without explicit class structure, and the goal is generally to construct a simple model which approximates the statistical behavior of $\mathbf{X}$.

The first setup that will be considered consists of the $n$-tuples $(\omega_1, \omega_2 ..., \omega_n)$, which are called *n-grams* in the context of language modeling. We are interested in developing statistical methods to estimate the conditional probabilities, $P(\omega_n|\omega_1 ..., \omega_{n-1})$, of observing these $n$-grams in a given application, in this dissertation in an ASR scenario. This probability greatly assists the ASR system in deciding upon one of possibly several acoustically-similar, competing ways of segmenting the observation vectors into words according to their linguistic likelihood. For instance, the ASR system could have difficulty in deciding upon the alternatives that include homonyms such as "two" and "to" in

> 'Oil prices fell down <u>to</u> $100 a barrel in <u>two</u> weeks."

For this purpose, we gather as many textual data as we can and extract all the $n$-grams. This typically results in millions of $n$-grams, each of which is observed $c(\omega_1, \omega_2 ..., \omega_n)$ of times in the data collected.

The second setup that we will consider is the classification problem for which the data samples consists of a random vector and a random variable, $(\mathbf{x}_i, y_i)_{i=1,...,N}$, where $\mathbf{x}_i$ is a *feature vector*, and $y_i$ is a *class label*. The feature vector is, for example, the spectral representation of a windowed speech signal, and the class label is the language spoken in this speech signal, e.g. ENGLISH, GERMAN, or FRENCH. We wish to determine the language, $y_i$, spoken in $\mathbf{x}_i$. The goal is then to learn a function, $y = f(\mathbf{x})$, such that it can predict the value of $Y$ for values of $\mathbf{X}$ that did not appear in the training set (e.g. a new spoken utterance belonging to one of the languages in the training set).

Starting from a set of data, the algorithms discussed can automatically extract a set of relationships inherent in the data and then combine these rules into a model of the data, which is both accurate and compact. For predicting a model and evaluating it afterwards, a statistical modeling problem involves several datasets:

- a *training set*, which is typically as large as possible and used to predict a model,
- a *test set* (or an evaluation set), which is application-specific and used to evaluate the resulting model,
- a *development set*, which is typically is a small set used to tune system parameters,
- an *adaptation set*, which is of limited amount, application-specific, and used to adapt an general model to a specific application.

In all cases, we have training and test sets but the other two sets are not always provided.

## *1.4 Formulating the Conflicting Objectives*

In this section, we analyze the two applications that we consider in this dissertation to unravel the conflicting objectives involved.

### 1.4.1 Conflicting Objectives in LM Adaptation for ASR

In ASR, the aim is to determine the most likely sequence of words, $W$, given the observed acoustic data, $\mathbf{x}$. This is achieved by finding that $\hat{W}$ that maximizes the conditional probability, $P(W|\mathbf{x})$. From the Bayes' rule,

$$\hat{W} = \arg\max_{W} P(W|\mathbf{x}) = \arg\max_{W} \frac{P(\mathbf{x}|W)P(W)}{P(\mathbf{x})} \tag{1.4.1}$$

but since $P(\mathbf{x})$ is constant for a given acoustic signal, an equivalent strategy is to find

$$\hat{W} = \arg\max_{W} P(\mathbf{x}|W)P(W). \tag{1.4.2}$$

The acoustic component of the speech recognizer must compute $P(\mathbf{x}|W)$, whereas the language model must estimate the prior probability of a given sequence of words, $P(W)$. Before focusing our attention on the latter, we will present some background on the acoustic modeling stage.

Central to every speech recognition system is a means of encoding the sounds comprising human speech. This has been most successfully achieved through the use of hidden Markov models (HMMs) [84, 112], although other approaches have also met with success [86]. By describing the observation vectors as a probabilistic time series, the HMM takes the inherent natural variability of human speech characteristics into account. Given a set of examples of a particular sound in the form of the corresponding observation vector sequences, $\mathbf{x}$, the parameters of the model, $\theta$, may be adjusted to best represent this data in a probabilistic sense, often by optimizing $P(\mathbf{x}|\theta)$. The model may consequently be employed to evaluate the likelihood of a new observation sequence with respect to its parameters, thus giving an indication of how similar the new measurement is to those originally used to determine its parameters.

This likelihood is used to make a statistical decision regarding the utterance to be recognized. HMMs may either be trained to model entire words directly or to model subword units (such as phonemes), which are concatenated to obtain words. The latter approach is usually adopted since, even for moderately-sized vocabulary, there may not be sufficient training material to determine whole-word models reliably.

While the acoustic model indicates how likely it is that a certain sequence of words matches the measured acoustic evidence, it is the task of the LM to estimate the prior probability of the word sequence itself. Let $W = \omega_1^N$ denote a word sequence with words $\omega_1$ through $\omega_N$, i.e., $W = \omega_1^N = \{\omega_1, \omega_2, ..., \omega_N\}$. Then, $P(W)$ may be decomposed as follows:

$$P(\omega_1^N) = \prod_{i=1}^{N} P(\omega_i|\omega_1^{i-1}) \tag{1.4.3}$$

Hence, the problem of assigning a probability to $W$ can be reduced to that of assigning a conditional probability to each word, $\omega_i$, in the word string given its word *history*, i.e., words $\omega_1$ through $\omega_{i-1}$, denoted as $\omega_1^{i-1}$. However, one cannot enumerate all possible word histories of any reasonable length. For this reason, word histories are partitioned into *equivalence classes*, $E_1, E_2, ..., E_m$, such that each possible word string $\omega_1^{i-1}$ belongs to one and only one equivalence class $E_i$. We then have

$$P(\omega_i|\omega_1^{i-1}) = P(\omega_i|E_i).$$

Statistical $n$-grams are the state-of-art LMs used in large vocabulary ASR systems [12, 42]. A *statistical n-gram LM* is a representation of an $(n-1)^{st}$ order Markov model in which the probability of the occurrence of a symbol is conditioned upon the occurrence of the preceding $(n-1)$ symbols, i.e., the equivalence classes are simply the preceding $(n-1)$ words. Such $n$-gram probability models are typically constructed from a large corpus of text by counting the co-occurrences of words in the vocabulary. Typically, every sentence is padded with a special symbol such as <s> at the beginning and with another special special symbol such as </s> at the end. For example, using

bigram language probabilities, the probability of the sentence

"Oil prices fell down to $100 a barrel."

would be calculated as

$$P(\text{oil}|<\text{s}>)P(\text{prices}|\text{oil})...P(\text{barrel}|\text{a})P(</\text{s}>|\text{barrel})$$

To estimate $P(\omega_i|\omega_{i-n+1}^{i-1})$, the frequency with which the word $\omega_i$ occurs given that the last $(n-1)$ words are $\omega_{i-n+1}^{i-1}$, we can simply count how often the bigram $\omega_{i-n+1}^{i}$ occurs in some text and normalize. Let $c(\omega_{i-n+1}^{i})$ denote the number of times the $n$-gram $\omega_{i-n+1}^{i}$ occurs in the given text. Then, we can take

$$P(\omega_i|\omega_{i-n+1}^{i-1}) = \frac{c(\omega_{i-n+1}^{i})}{\sum_{\omega_i} c(\omega_{i-n+1}^{i})} \tag{1.4.6}$$

The estimate for $P(\omega_i|\omega_{i-n+1}^{i-1})$ given in (1.4.6) is called the *maximum likelihood* (ML) estimate of $P(\omega_i|\omega_{i-n+1}^{i-1})$, because this assignment of probabilities yields the $n$-gram model that assigns the highest probability to the training data of all possible $n$-gram models. In many applications, $n = 3$ is used, and this model is referred to as a trigram model.

The key difficulty with using $n$-gram LMs, as well as many of the alternatives, is that of *data sparsity*. One can never have enough training data to estimate all of the model's parameters reliably. The language modeling typically requires millions of probabilities to be estimated. Luckily, most current ASR systems are dedicated to one specific task (for example, the recognition of broadcast news). Such text domains may be distinguished by attributes such as the topic of discussion, style of writing, and the time of writing. A model trained on text from many diverse styles of language might perform better in general, but will not be especially well suited to any particular application domain. On the other hand, an application-specific LM will often perform very badly on language from a different domain. For example, an

LM trained with weather forecasts would not work well predicting test data about financial news, no matter how much weather forecast texts are available. Moreover, often there are insufficient data from the application domain to allow specialized models to be built directly. Thus the idea of an adaptive language model whose parameters automatically adjust to the current style of language is an appealing one [88]. The task of LM adaptation involves a detail versus specificity tradeoff concerning how much we should depend on either domain.

### 1.4.2 Conflicting Objectives in Automatic Spoken Language Identification

Automatic language identification (LID) is the problem of identifying the language being spoken by an unknown speaker from a sample of speech. It is often used as a front-end system to a language-specific speech recognition system for applications such as directory assistance, machine translation, and multi-lingual information retrieval.

Let $L = \{L_1, ..., L_M\}$ be the set of the target languages that a system is designed to identify. The classification of a given acoustic signal $\mathbf{x}$ into a language $L^*$ is then given by

$$L^* = \arg\max_L P(L|\mathbf{x}) \cdot P(L). \tag{1.4.7}$$

Most often, the prior probability of the target languages is assumed uniform and (1.4.7) is equivalent to

$$L^* = \arg\max_L P(L|\mathbf{x}). \tag{1.4.8}$$

It is possible to develop an LID system based only on the information that is directly available from the acoustic signal, $x$. Such an approach directly implements (1.4.8). More often, however, the acoustic signal, $\mathbf{x}$, is first converted into a string of linguistic units, such as phones or words. The resulting string is further analyzed in terms of linguistic clues as to which language they most likely belong to. Languages have characteristic sound patterns and differ in the inventory of phonological units (speech

sound categories) used to produce words, the frequency of the occurrences of these units, and the order in which they occur in words. The performance of an LID system can be measured by a number of criteria. Among these, the miss and false-alarm rates are the most common.

## *1.5 Contributions and Organization*

The objective of this thesis is to develop new MOP-based algorithms for problems with competing objectives and to understand the behavior of the proposed algorithms in light of some applications. Furthermore, we strive to analyze the properties of the developed algorithms, including convergence and optimality, in a mathematical framework.

This dissertation makes contributions to the problem of learning parameters of a statistical model from the following three aspects:

- This dissertation investigates the use of MOP in speech and language applications, which is not well-explored. MOP is in fact well-suited for similar applications because
  - incommensurate objectives can easily be handled,
  - the range of obtainable objective functions is wider and no opportunity is missed,
  - the solution methodologies are intuitive.
- This dissertation is expected to provide new insights and techniques for accomplishing significant performance improvement over existing approaches in terms of the individual competing objectives. Meantime, the designer has a better control over what is achieved in terms of the individual objectives.
- Most of the research attempts that aim at using MOP in similar applications

illustrate their methods in problems with two or three objectives. This dissertation reports experimental results for as many as 30 conflicting objectives in LID and four objectives in statistical LM adaptation.

Our discussion will proceed as follows. Chapter II covers the foundations, which will be used in the following chapters. It starts with the basic concepts in information theory, including entropy and Kullback-Leibler divergence. Finally, some methods commonly used in speech and language applications are discussed. This is followed by an overview of single- and multi-objective programming in Chapter III. The concept of Pareto optimality and MOP methods are briefly described. Chapter IV discusses the work related to the applications considered in this dissertation. Chapter V describes our novel MOP-based technique, called iterative constraint optimization (ICO), and its properties.

Chapter VI discusses the application of estimating the $n$-gram probabilities to be used in an ASR task. We start with a novel SOP-based technique, called structural maximum a posteriori (SMAP) method, in which each $n$-gram of interest is embedded in a branch of a tree structure. The nodes in the first layer of such trees represent the unigrams, and those in the second layer represent the bigrams, and so on. By modeling the prior distribution with a Dirichlet distribution, we obtain a recursive formula for estimating the hyperparameters of the prior distribution in a top-down manner. The Bayesian formulation yields a closed form solution for the $n$-gram probabilities, which combines the counts of $n$-gram events and the weighted prior density hyperparameters.

The LM adaptation approach developed in the continuation of Chapter VI is based on reformulating the training objective of the structural MAP (SMAP) method that we proposed in the first part as an multi-objective problem. We extract the individual *at least partially conflicting* objective functions in the SMAP formulation. For a bigram LM, this yields a problem with four objectives: The first two objectives are

concerned with the best fit to the adaptation data while the remaining two objectives are concerned with the best prior information obtained from a general domain corpus. Solving this problem in an iterative manner such that each objective is optimized one after another with constraints on the rest, we obtain a target LM that is a log-linear interpolation of the component LMs. The LM weights are found such that all the (at least partially conflicting) objectives are optimized simultaneously.

Chapter VII discusses an SOP- and an MOP-based approaches to LID problem. As in Chapter VI, we start with an SOP-based approach. This SOP-based approach is based on minimizing the average of different error rates, namely, the false-acceptance (or false-alarm) and false-rejection (or miss) rates. The error rates are formulated as differentiable functions of system parameters by using the minimum classification error rate (MCE) framework. This is followed by the development of an MOP-based approach in which each error rate is treated as an individual objective function. The goal is to reduce all of the error rates into comparable levels that are as small as possible. This is achieved by optimizing the error rates one after another in an iterative manner. Finally, Chapter VIII discusses the major research results and future research directions.

# CHAPTER II

# FOUNDATIONS

In this chapter, the basic concepts and methods that will be used in the continuation are reviewed. We will start with defining fundamental concepts from information theory that will be used when developing an MOP approach to language modeling and adaptation problem. This will be followed with the description of some methods that are used in speech and language applications for reliable estimating model parameters.

## *2.1 Fundamentals of Information Theory*

Information theory was introduced by Claude Shannon [94] as a mathematical theory of communication. The fundamental problem addressed by Shannon was how to transmit a source over a noisy channel such that it can be reconstructed with acceptable error, while making minimum use of the channel.

One of Shannon's basic insights was that most communication channels can be broken into two components, as shown in Figure 3. The first is the source $\mathbf{X}$, which one wants to transmit. For instance, $\mathbf{X}$ may represent a speech signal that one speaks. The distribution of the source is denoted by $P(\mathbf{x})$. A low $P(\mathbf{x})$ indicates that $\mathbf{x}$ will only rarely be sent. The second component is a noisy channel, over which the transmission is carried out. The output, $Y$, of the channel is usually a stochastic function of the input, and its behavior can be described via a distribution $P(y|\mathbf{x})$. Here, the output $y$ may several interpretations. First, $y$ can be the received speech signal, which will typically different from the original speech signal, $\mathbf{x}$, since $\mathbf{x}$ undergoes some degradation as a result of noise, e.g., a low-quality telephone line. Second, $y$ can also be a label assigned to the speech signal, $\mathbf{x}$, after appropriate processing. For instance, $y$ may be a label from the label space $\mathcal{Y}$ ={MALE SPEECH, FEMALE SPEECH}, or a label


**Figure 3:** The noisy channel model of the transmission of information.

from another label space , $\mathcal{Y}$ ={ENGLISH, FRENCH, KOREAN}.

*Entropy*, denoted as $H$, is the degree of uncertainty one has about the variable, **X**, before observing it. The fundamental coding theorem of information theory states that on average $H$ bits are required to represent a symbol emitted from a source of entropy $H$ [95].

**Definition 2.1.1** *The entropy of a discrete random variable, $X$,* $\mathrm{H}(X)$ *is defined by*

$$\mathrm{H}(X) = -\sum_{\omega} P(x) \log P(x). \tag{2.1.1}$$

In this thesis, information theory will be useful in the context of language modeling, which are discrete conditional probabilities of words, $\omega$, given their history, $h_\omega$, i.e., the probabilities $P(\omega|h_\omega)$. The information provided by a source emitting words is measured by its *conditional entropy*, which is the average information provided by any word, $\omega$, averaged across all possible histories, $h_\omega$.

**Definition 2.1.2** *The conditional entropy* $\mathrm{H}(W|h(W))$ *is defined by*

$$\mathrm{H}(W|h(W)) = -\sum_{h_\omega} P(h_\omega) \sum_{\omega} P(\omega|h_\omega) \log P(\omega|h_\omega). \tag{2.1.2}$$

Another useful concept from information theory is the *Kullback-Leibler (KL) divergence*, also known as the relative entropy or the discriminant information [57]. It is a measure of the separation between two distributions [20].

**Definition 2.1.3** *The Kullback-Leibler (KL) divergence between two probability mass functions, $P(X)$ and $Q(X)$, is defined as*

$$D(P||Q) = \sum_{x} P(X) \log \frac{P(X)}{Q(X)}. \tag{2.1.3}$$

The KL divergence is zero if and only if $P = Q$. Even though the KL divergence is not a distance since it is not symmetric and does not satisfy the triangular inequality, it is often useful to think of the KL divergence as a distance between probability distributions, or a measure of the inefficiency of assuming that the distribution is $Q$ when the true distribution is $P$. In the context of $n$-gram LMs, this distance is measured with conditional KL divergence.

**Definition 2.1.4** *The conditional KL divergence* $D(P(\omega|h_\omega)||Q(\omega|h_\omega))$ *is given by*

$$D(P(\omega|h_\omega)||Q(\omega|h_\omega)) = \sum_{h_\omega} P(h_\omega) \sum_{\omega} P(\omega|h_\omega) \log \frac{P(\omega|h_\omega)}{Q(\omega|h_\omega)}. \tag{2.1.4}$$

## *2.2 Methods in Speech and Language Technologies*

The advances in speech and language technologies have become possible after much progress have been done in statistical modeling. In this section, some of the most important statistical techniques that will be used in later chapters are reviewed.

### 2.2.1 Generative vs. Discriminative Models

Given input data point $\mathbf{x}$, a *discriminative model* computes $P(y|\mathbf{x}), y = \mp 1$, i.e., the probability of $\mathbf{x}$ being positive or negative. We only need to compute $P(y = +1|\mathbf{x})$ since $P(y = -1|\mathbf{x}) = 1 - P(y = +1|\mathbf{x})$. A *generative model*, on the other hand, often captures the generation process of $\mathbf{x}$ by modeling $P(\mathbf{x}|y = +1)$ and $P(\mathbf{x}|y = -1)$.

Generative models have long been used in speech, text, and vision for their ability to handle variable-length structured data. For example, an HMM with a fixed set of parameters can generate an observation sequence of arbitrary length. Moreover, generative models have principled ways to treat latent variables, typically using the expectation-maximization algorithm [23]. Despite these advantages, this approach is in general sub-optimal for classification tasks, as it intends to solve a more difficult density estimation problem rather than to optimize directly for classification performance.

Discriminative models, on the other hand, directly model the conditional relationship of labels given input features. Discriminative models are mostly focused on how well they can separate the positive samples from the negative samples. Log-linear models, multi-layer perceptrons (MLPs), conditional maximum entropy (MaxEnt) models, and conditional random fields (CRFs) all belong to probabilistic discriminative models.

A second class of discriminative models directly model the decision boundary, which is the most relevant information to classification. An affine decision function, for example, assumes that the decision boundary is a hyperplane. Moreover, if applied in a transformed feature space, this approach can as well model nonlinear decision boundaries. In such cases,

$$y = sign(\mathbf{w}^T \phi \mathbf{x} + b) \tag{2.2.1}$$

where $\mathbf{w}$ and $b$ are affine parameters, $\mathbf{w}^T$ represents the transpose of $\mathbf{w}$, and $\phi(\mathbf{x})$ is a nonlinear transformation. Non-probabilistic discriminative models include kernel methods such as support vector machines and nearest neighbor methods.

A *discriminant* is a function, which is mathematically denoted by $g(\tilde{\mathbf{x}}, \mathbf{w})$, that takes an input vector, $\mathbf{x}$, and assigns it to one of $K$ classes. For instance, a set of $M$ linear discriminant functions (LDFs) are defined as [24]

$$g(\mathbf{x}, \mathbf{w}_m) = \mathbf{w}_m^T \mathbf{x}, \mathbf{w}_m \in \Re^n, m = 1, ..., M, \tag{2.2.2}$$

where $\mathbf{w}_m^T$ represents the transpose of the weight vector of the $m^{th}$ class. An unseen sample, $\tilde{\mathbf{x}}$, is assigned to the class $\hat{C}$ for which $g(\tilde{\mathbf{x}}, \mathbf{w}_j)$ is maximized, i.e,

$$\hat{C} = \arg\max_j g(\tilde{\mathbf{x}}, \mathbf{w}_j). \tag{2.2.3}$$

### 2.2.2 Parameter Learning Methods

#### *2.2.2.1 Minimum Classification Error Rate (MCE) Training*

Let $\Theta$ denote a set of the weight vectors of $M$ discriminant functions, i.e., $\Theta = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$. To simulate the cost of the decision function in (2.2.3), a class-specific misclassification function, $\pi_m(\mathbf{x}, \theta)$, is assigned to each sample as

$$\pi_m(\mathbf{x}, \Theta) = -g(\mathbf{x}, \mathbf{w}_m) + \log[\frac{1}{M-1}\sum_{i \neq m} \exp(\eta g(\mathbf{x}, \mathbf{w}_i))]^{\frac{1}{\eta}}, \tag{2.2.4}$$

where $\eta$ is a positive number [17, 47]. To approximate a given objective criterion as a function of $\Theta$, $\pi_m(\mathbf{x}, \Theta)$ is embedded in a sigmoid function as

$$\ell_m(\mathbf{x}, \Theta) = \frac{1}{1 + \exp(-\alpha \pi_m(\mathbf{x}, \Theta) + \beta_m)}. \tag{2.2.5}$$

Let $C_m$ denote the set of the training samples that are labeled as samples of the $m^{th}$ language. For an $m^{th}$ class sample, $\pi_m(\mathbf{x}, \Theta) > 0$ implies misclassification, and $\pi_m(\mathbf{x}, \Theta) \leq 0$ implies correct classification. Clearly, when $\pi_m(\mathbf{x}, \Theta)$ is very small, which implies correct classification, $\ell_m(\mathbf{x}, \Theta)$ is very small and virtually no loss is incurred. When $\pi_m(\mathbf{x}, \Theta)$ is very large, it leads to a penalty that essentially becomes a classification/recognition error count.

The parameters $\alpha$ and $\beta$ specify the slope and intersect of the sigmoid function. As $\alpha$ gets larger, the sigmoid gets steeper. As $\beta$ gets larger, the sigmoid is shifted. This is illustrated in Figure 4. The region where the sigmoidal function is non-saturated is called the learning window. The samples falling in this region contribute to the parameter learning.

#### *2.2.2.2 Maximum A Posteriori Principle*

It is often desired to determine the best hypothesis from some space, $Y$, given the training data samples, $\mathbf{x}$. One way to specify what we mean by the best hypothesis is to say that we demand the most probable hypothesis, given the data, $\mathbf{x}$, plus any initial knowledge about the prior probabilities of the various hypotheses in $Y$. Bayes'

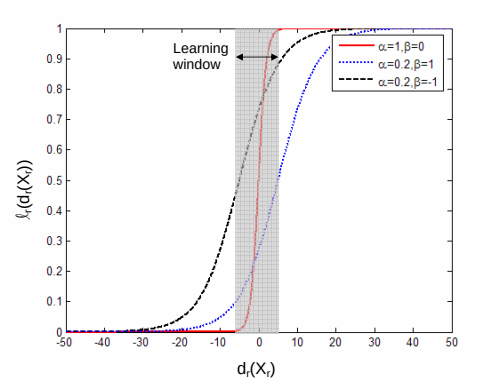



**Figure 4:** The sigmoid function approximating 0-1 loss function.

theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself. It is given as

$$P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}. \tag{2.2.6}$$

The probability $P(y)$ denotes the initial probability that the hypothesis $y$ holds before we have observed the data. It is called the *prior probability* of $y$ and reflects any background knowledge about $y$. Likewise, the probability $P(\mathbf{x})$ denotes the prior probability that the data, $\mathbf{x}$, will be observed given no information regarding which hypothesis holds true. The probability $P(y|\mathbf{x})$ denotes the chance that $h$ holds true given the observed data $\mathbf{x}$. It is called the *posterior probability* of $y$ and reflects our confidence that $y$ holds after we have seen the data, $\mathbf{x}$.

In many learning scenarios, the learner considers a set of candidate hypotheses, $Y$, and wants a find the maximally probable hypothesis, which is called a *maximum a posteriori* (MAP) hypothesis, $y_{MAP}$. The MAP hypothesis can be determined by using the Bayes theorem to calculate the posterior probability of each candidate hypothesis, i.e.,

$$y_{MAP} = \arg\max_{y \in Y} P(y|\mathbf{x}) = \arg\max_{y \in Y} \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} = \arg\max_{y \in Y} P(\mathbf{x}|y)P(y) \tag{2.2.7}$$

#### *2.2.2.3 Maximum Likelihood Principle*

When we do not have any prior knowledge about $y$, we might simply assign the same probability to each candidate hypothesis. In this case, we can simplify (2.2.7) and consider the term $P(\mathbf{x}|y)$ to find the most probable hypothesis. $P(\mathbf{x}|y)$ is called the *likelihood* of the data, $x$, given $y$, and the hypothesis that maximizes $P(\mathbf{x}|y)$ is called a *maximum likelihood* (ML) hypothesis, $y_{ML}$, i.e.,

$$y_{ML} = \arg\max_{y \in Y} P(\mathbf{x}|y) \tag{2.2.8}$$

The ML estimation is the most widely used parametric estimation method mainly because of its efficiency. It is assumed that the parameters of the involved probability distributions are fixed but unknown and aims to find the set of parameters that maximizes the likelihood of the observed data.

#### *2.2.2.4 Maximum Entropy Principle*

Suppose one knows that a distribution $P(\mathbf{x})$ has a known expected, what can be said about the values of $P(\mathbf{x})$ for all $\mathbf{x}$? One of the key approaches to this problem has been the Maximum Entropy (ME) principle introduced in statistical mechanics in the 19th century and later expanded by Jaynes [41].

Suppose that we are given $N$ binary-valued feature functions, $f_i$, which determine the statistics we feel are important in modeling the process. We want our model to accord with these statistics. That is, we want our model to lie in the subset $\mathcal{P}$ of defined by

$$\mathcal{P} = \{P(y|\mathbf{x})| \sum_{\mathbf{x},y} \tilde{P}(\mathbf{x})P(y|\mathbf{x})f_i(\mathbf{x},y) = \sum_{\mathbf{x},y} \tilde{P}(\mathbf{x},y)f_i(\mathbf{x},y), fori = 1,2,...,n\} \tag{2.2.9}$$

where $\tilde{P}$ is an empirical distribution. Among the models $P \in \mathcal{P}$, the maximum entropy philosophy dictates that we select the most uniform distribution. A mathematical measure of the uniformity of a conditional distribution is provided by the

conditional entropy

$$H_P(Y|\mathbf{X}) = -\sum_{\mathbf{x}} \sum_{y} \tilde{P}(\mathbf{x}) P(y|\mathbf{x}) \log P(y|\mathbf{x}). \tag{2.2.10}$$

An optimization method specifically tailored to the maximum entropy problem is the iterative scaling algorithm of Darroch and Ratcliff [22]. With this definition in hand, the principle of maximum entropy is stated as follows:

To select a model from a set, $\mathcal{P}$, of allowed probability distributions, choose the model, $p^*$, with maximum entropy, $H_P(Y|\mathbf{X})$:

$$P^* = \arg\max_{P \in \mathcal{P}} H_P(Y|\mathbf{X}) \tag{2.2.11}$$

Solving this problem with Lagrangian function method, we obtain an analytical solution for $P^*(y|\mathbf{x})$ as

$$P^*(y|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_i \lambda_i f_i(\mathbf{x}, y)\right) \tag{2.2.12}$$

where $Z(\mathbf{x})$ is a normalization factor.

The ME distribution is intuitively the distribution with the highest degree of uncertainty and as such reflects no additional knowledge above that given in the observations (it is sometimes referred to as the distribution least committed to unseen data).

#### *2.2.2.5 Minimum Discrimination Information Principle*

When $Q(\mathbf{x})$ is the uniform distribution, the KL divergence is the negative entropy of $P(\mathbf{x})$ (up to an additive constant), and thus KL divergence is equivalent to ME. The distribution $Q(\mathbf{x})$ can be interpreted as supplying some prior knowledge about the distribution of $\mathbf{X}$.

Kullback's minimum discrimination information (MDI) principle [56] generalizes this concept. It seeks to minimize the KL divergence $D(P, Q)$, which means it seeks to determine the distribution $P$ that satisfies all the constraints and is closest to a given

distribution $Q$. If we have the same constraints as in (2.2.9), the optimal probabilities are found to be

$$P^*(y|\mathbf{x}) = \frac{Q(y|\mathbf{x})}{Z(\mathbf{x})} \exp\left(\sum_i \lambda_i f_i(\mathbf{x}, y)\right) \tag{2.2.13}$$

Minimizing KL divergence of a target model from a non-informative prior (i.e., from a uniform prior) is equivalent to maximizing the likelihood [10]. When there is reliable prior information, typically another term is included in addition to maximizing the likelihood. Suppose that the data, $\mathbf{X}$, comes from an unknown distribution, $P(\mathbf{x})$, that we wish to model. We can approximate this distribution using some parametric distribution $Q_\theta(\mathbf{X})$, governed by a set of adjustable parameters, $\theta$. We can determine $\theta$ by minimizing the KL divergence between $P(\mathbf{x})$ and $Q_\theta(\mathbf{x})$ with respect to $\theta$. Given the training samples $\mathbf{x}_t$, $t = 1, ..., T$, the expectation with respect to $P(\mathbf{x})$ can be approximated by a finite sum over these points so that

$$D(P \parallel Q) \simeq \sum_{t=1}^{T} -\log q_\theta(\mathbf{x}_t) + \log p(\mathbf{x}_t), \tag{2.2.14}$$

where we used the approximation

$$\mathbb{E}[f] \simeq \frac{1}{T} \sum_{t=1}^{T} f(\mathbf{x}_t).$$

Note that the first term in (2.2.2.5) is the negative log-likelihood function for $\theta$ under the distribution $q_\theta(\mathbf{x})$ evaluated using the training set. Meantime, the second term on the right-hand side of does not depend on $\theta$. Thus, maximizing the likelihood function is equivalent to minimizing the KL divergence of the target model from a model obtained from the in-domain data, i.e., to minimizing $D(P||Q_\theta)$ [10].

In [41], Jaynes advocated the use of maximum entropy priors in Bayesian decision theory since "It agrees with what is known, but expresses a maximum uncertainty with respect to all other matters, and thus leaves a maximum possible freedom for our final decisions to be influenced by the subsequent sample data." Suppose that we have some reliable source to obtain prior information. This source enables us to

derive a prior distribution. Then, minimizing the KL divergence of the target model from the prior model makes the target model *spread out as uniformly as possible without contradicting the given information.*

In a MAP formulation, it is convenient if the prior distribution is chosen from a *conjugate family*, i.e., a family closed under conditioning on observed data. This means that the prior and the posterior probability distributions belong to the same family, and the only changes made after observing data are in the model parameters, not in the model itself. The conjugate prior density is the distribution which minimizes the KL divergence of the target posterior model from the prior distribution, i.e., which minimizes $D(P||Q_\theta)$ [6]. Minimizing the KL divergence of the target model from the prior model makes the target model spread out as uniformly as possible without contradicting the given information [41]. Thus, the use of conjugate prior density implicitly minimizes $D(P||Q_\theta)$.

### 2.2.3 Regularization

Since the available sample size in a given training set is almost always far from sufficient, the use of certain regularization strategy is required to guarantee good generalization performance [2, 65, 92, 105].

One general approach of regularization is to make a new criterion function that depends not only on the classical training error but also on model complexity. Specifically, the new criterion function penalizes highly complex models; searching for the minimum in this criterion is to balance error on the training set with complexity. Formally, we can write the new error as the sum of the error in the training set, $f_{error}$, *plus* a regularization term, $f_{reg}$, which expresses constraints or desirable properties of solutions as

$$f = f_{error} + \alpha f_{reg} \tag{2.2.15}$$

The parameter $\alpha$ is adjusted to impose the regularization more or less strongly. The

simplest penalty term takes the form of a sum of squares of all of the coefficients as

$$f_{reg} = \|\mathbf{w}\|^2 = \sum_{i=1}^{n} w_i^2 \tag{2.2.16}$$

### 2.2.4 Evaluation Criteria

#### *2.2.4.1 Evaluation of Language Modeling and Adaptation Techniques*

The utility of an LM is intrinsically linked with the effect which it has on the accuracy of the application in which it is used. When used for ASR application, this accuracy is generally measured in terms of the *word error rate* (WER), which is defined as the total number of errors (word insertions, deletions, and substitutions) divided by the total number of words actually said, i.e.,

$$\text{WER} = \frac{\#\text{ insertions} + \#\text{ deletions} + \#\text{ substitutions}}{\#\text{ words}}. \tag{2.2.17}$$

Recognition experiments are computationally costly, however, and so other methods of evaluating the quality of an LM are typically used. The most common of these is the *perplexity*, which is based on the concepts of information theory.

General sources (such as language) will not output their symbols independently. For a source, $W$, emitting symbols, $\omega_1, \omega_2, ..., \omega_n$, the entropy is

$$H(W) = -\lim_{n\to\infty} \frac{1}{n} \sum_{\omega_1,\omega_2,...\omega_N} P(\omega_1, \omega_2, ..., \omega_N) \log_2 P(\omega_1, \omega_2, ..., \omega_N). \tag{2.2.18}$$

If the information source is assumed ergodic and if $N$ is sufficiently large, $H(W)$ can be approximated as

$$H(W) \approx -\frac{1}{N} \log_2 P(\omega_1, \omega_2, ..., \omega_N). \tag{2.2.19}$$

Let $\tilde{W} = \tilde{\omega}_1^N$ be an unseen word sequence. Then, as far as the ASR system is concerned, a measure of the difficulty of the speech recognition process is given by

$$H(\tilde{W}) = -\frac{1}{N} \log_2 P(\tilde{\omega}_1^N) \tag{2.2.20}$$

If $N$ is sufficiently large, and the language source is ergodic, then it will always be the case that $H \geq \hat{H}$ . Thus the lowest possible value of $H(\tilde{W})$ can only be achieved

using a perfect model of the language. Therefore, a good model is one which has a low entropy, and hence assigns a high probability to the test text. The perplexity (PP) is given as follows:

$$PP = 2^{H(\tilde{W})} = P(\tilde{\omega}_1^N)^{-\frac{1}{N}} \tag{2.2.21}$$

A low perplexity does not mean that the LM is good in the sense of leading to accurate ASR systems. If one is concerned with reducing the number of errors made by ASR systems, she/he should also consider the acoustic similarities between words: An LM that is capable of discriminating between acoustically similar words would be more useful than one which is not. Furthermore, the accuracy of an ASR system will depend not only on the probabilities assigned to the correct hypothesis, but also on the probabilities assigned to all other candidate hypotheses which will be competing with it.

#### *2.2.4.2 Evaluation of LID Techniques*

The most obvious measure of LID system performance is the system's ability to reliably identify the language of a spoken utterance. An examination of the receiver-operator characteristic (ROC) curve [29] of the system can help determine the reliability of the system's scoring mechanism. The ROC is calculated by setting a threshold on the system's top-choice score and rejecting all utterances that fall below that threshold. The threshold is varied to examine the system's performance as the rejection region is spanned from 0% rejection to 100% rejection.

It has been found useful in speech applications to use a variant of ROC called the detection error tradeoff (DET) curve [67]. In a DET curve, the abscissa axis shows the false-alarm rate while the ordinate axis shows the detection rate on linear scales. The DET curves are almost linear, and they permit an easy observation of system contrasts since they are more spread out than ROC curves.

Aside from reliability, an LID system may also be evaluated based on other aspects

such as its computational requirements, required training set size, and portability to different language sets.

**NIST Language Recognition Evaluations**

The National Institute of Standards and Technology (NIST) coordinated language recognition evaluations (LREs) in 1996, 2003, 2005, and 2007. The design of the LID systems developed in this thesis are based on the NIST specifications for LRE 2005.

The NIST LREs were intended to establish a current baseline of performance capability for language and dialect recognition of conversational telephone speech and to stimulate further research efforts in the field. Given a test segment of speech and a target language or dialect, the system to be evaluated was to determine whether or not the speech is from the target language or dialect.

The target languages and dialects included ENGLISH (American), ENGLISH (Indian), HINDI, TAMIL, JAPANESE, KOREAN, MANDARIN (Mainland), MANDARIN (Taiwan), and SPANISH (Mexican). In addition to these target languages, an additional category "OTHER" was included as a possible language class for test data.

The performance of a detection system in NIST LREs is characterized by its miss and false-alarm probabilities. Performance for detection for a target language, $L_i$, was measured using a detection cost function, $C_{DET(i)}$, which represents the expected cost of making a detection decision. It is calculated for each of the $M$ languages separately and is defined as

$$C_{DET}(i) = \frac{1}{2}P(\text{Miss}(i)|\text{Target}) + \frac{1}{M-1}\sum_{j\neq i}\frac{1}{2}P(\text{False-alarm}(i)|\text{Non-Target}(j)) \tag{2.2.22}$$

where $P(\text{Miss}(i)|\text{Target})$ is the computed percentage of misses for target trials involving $L_i$, and $P(\text{False-alarm}(i)|\text{Non-Target}(j))$ is the computed percentage of false alarms for non-target trials involving detection of $L_i$ when the test segment language

is $L_j$. The overall average detection cost function is

$$C_{DET} = \frac{1}{M} \sum_i C_{DET}(i) \tag{2.2.23}$$

For each test segment, all relevant language and dialect hypotheses were applied in turn. Thus there was a total of $M$ different trials for each test segment. For each trial, each system under test provided two outputs. The first output is the actual decision (TRUE or FALSE) of whether or not the language/dialect spoken in the test segment is the target language/dialect. The second output is a score indicating the relative likelihood that the language/dialect of the test segment is the target language/dialect.

# CHAPTER III

# MATHEMATICAL PROGRAMMING

MOP problems are usually solved by scalarization, i.e., the problem is converted into a set of single-objective programming (SOP) problems. After scalarization, the methods developed for single-objective problems are used.

The first part of this chapter is devoted to an overview of the SOP techniques used in the MOP algorithms proposed in Chapters 5 and 6. In the second part, the optimality conditions and solution methodologies for multi-objective problems are briefly described.

## *3.1 Single-Objective Programming (SOP)*

Suppose that we want to choose a set of $M$ decision vectors, denoted as $\Theta = \{\mathbf{w}_1, ..., \mathbf{w}_M\}$. Each of the decision vectors in an $n$-dimensional vector, i.e., $\mathbf{w}_m \in \Re^n$, $m = 1, ..., M$. Single objective programming (SOP) is a branch of mathematical programming in which a single-valued objective function $f$ of the unknowns, is minimized (or maximized).

### 3.1.1 Unconstrained Nonlinear Optimization

In unconstrained optimization, the objective function depends on a set of real variables with no restriction on the values of the decision variables. The mathematical formulation is simply

$$\min_{\mathbf{w}\in\Re^n} f(\mathbf{w}), \tag{3.1.1}$$

where $f : \Re^n \rightarrow \Re$ is a smooth function. For minimality at a point, $\mathbf{w}^*$, two conditions need to be satisfied:

$$(1) \qquad \nabla f(\mathbf{w}^*) = 0,$$

$$(2) \qquad \nabla^2 f(\mathbf{w}^*) \text{ is positive semi-definite}$$

where $\nabla f(\mathbf{w}^*)$ is the gradient of $f$ at the optimal solution $x^*$, and $\nabla^2 f(\mathbf{w}^*)$ is the Hessian matrix of $f$ at $\mathbf{w}^*$. Beginning at an initial point $\mathbf{w}_0$, a numerical optimization algorithm generates a sequence of iterates, $\{\mathbf{w}_k\}_{k=0}^{\infty}$, that terminates when no more progress can be made.

The goal in deciding how to move from one iterate, $\mathbf{w}_k$, to the next iterate, $\mathbf{w}_{k+1}$, is to reduce $f$ as much as possible. One of the commonly used technique to find a direction in which the objective $f$ is reduced is the Newton's method [7, 80]. Despite its quadratic rate of convergence, the Newton's method has several drawbacks, and therefore needs to be converted into a reliable minimization algorithm. *Quasi-Newton methods* are descent methods with the quasi-Newton direction

$$\mathbf{d}_k = -H_k \nabla_{\mathbf{w}} f(\mathbf{w}_k), \tag{3.1.2}$$

where $H_k$ is a positive-definite matrix adjusted during the course of the computation in a way that (3.1.2) approximates Newton's method. Despite a large variety of quasi-Newton methods, there is growing evidence that the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method is the best general-purpose quasi-Newton method currently available [7, 80]. In BFGS algorithm, $H_{k+1}$ is obtained from $H_k$ by means of the equation

$$H_{k+1} = H_k + \left(1 + \frac{\gamma_k' H_k \gamma_k}{\delta_k' \gamma_k}\right) \frac{\delta_k \delta_k'}{\delta_k \gamma_k} - \frac{\delta_k \gamma_k' H_k + H_k \gamma_k \delta_k'}{\delta_k' \gamma_k}, \tag{3.1.3}$$

where $\delta_k = \mathbf{w}_{k+1} - \mathbf{w}_k$ and $\gamma_k = \nabla f(\mathbf{w}_{k+1}) - \nabla f(\mathbf{w}_k)$. The BFGS algorithm generates a sequence of solutions, $\{\mathbf{w}_k\}_{k=0}^{\infty}$, that converges to $\mathbf{w}^*$ at a super-linear convergence rate [7].

Once a direction, $\mathbf{d}_k$, is determined, $\mathbf{w}_{k+1}$ is obtained from $\mathbf{w}_k$ as follows:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \mathbf{d}_k, \tag{3.1.4}$$

where $\alpha_k$ is known as the *step size* ate the $k^{th}$ iteration. There are a number of rules for choosing $\alpha_k$. One of them widely used in practice is the *Armijo rule* [7,80], which generates solutions $x_k$ that converge to $x^*$ at a super-linear convergence rate.

### 3.1.2 Constrained Nonlinear Optimization

Two classical nonlinear programming problems are the equality constrained problem

$$\begin{aligned} \min \quad & f(\mathbf{w}) \\ \text{subject to} \quad & c_i(\mathbf{w}) = 0,\ i \in \xi, \end{aligned} \tag{3.1.5}$$

and its inequality constrained version

$$\begin{aligned} \min \quad & f(\mathbf{w}) \\ \text{subject to} \quad & c_i(\mathbf{w}) \geq 0,\ i \in \Im, \end{aligned} \tag{3.1.6}$$

where $f : \Re^n \to \Re$ and $c_i : \Re^n \to \Re^m$ are given functions.

The Karush-Kuhn-Tucker (KKT) optimality conditions for the inequality-constrained problem in (3.1.6) can be written as [7, 80]

$$(1) \quad \nabla L(x^*, \lambda^*) = 0,\ \textit{Optimality} \tag{3.1.7}$$

$$(2) \quad c_i(x^*) \geq 0,\ i \in \Im,\ \textit{Feasibility} \tag{3.1.8}$$

$$(3) \quad \lambda_i^* c_i(x^*) = 0,\ i \in \Im,\ \textit{Complementary Slackness} \tag{3.1.9}$$

$$(4) \quad \lambda_i^* \geq 0,\ i \in \Im \tag{3.1.10}$$

As an interpretation of KKT conditions, note that $c_i(x) = 0$ means that $x$ is *feasible* and the $i^{th}$ constraint is *active* (*binding*). Then complementary slackness condition (3.1.9) requires that $\lambda_i \neq 0$ where $\lambda_i > 0$ by (3.1.10). If $c_i(x)$ becomes negative, the point $x$ is *infeasible*, and if $c_i(x)$ becomes positive, $x$ is *feasible* and the

$i^{th}$ constraint is inactive (not binding). In the latter case, complementary slackness condition (3.1.9) requires that $\lambda_i = 0$. Feasibility condition (3.1.8) prohibits $c_i(x)$ from being negative at the optimal solution. Lastly, optimality condition requires the gradient $\nabla L(x, \lambda)$ be 0 at the optimal solution $(x^*, \lambda^*)$.

A fundamental approach to constrained optimization problem in (3.1.5) is to replace the original constrained problem by a sequence of unconstrained subproblems. In this section we discuss three approaches in this class. The first approach is the *(ordinary) Lagrangian function method*

$$L(\mathbf{w}, \lambda) = f(\mathbf{w}) - \sum_{i \in \xi} \lambda_i c_i(\mathbf{w}), \tag{3.1.11}$$

where $\lambda_i$'s are the unknown Lagrangian parameters. The (ordinary) Lagrangian function method has some fundamental limitations, mainly the lack of a good mechanism to enforce convergence when far from a solution.

A second approach that is widely accepted is the class of *penalty methods*. The *quadratic penalty function* is given as

$$Q(\mathbf{w}, \rho) = f(\mathbf{w}) + \frac{\rho}{2} \sum_{i \in \xi} c_i^2(\mathbf{w}) \tag{3.1.12}$$

where $\rho$ is a penalty parameter. The penalty parameter sequence, $\{\rho_k\}_{k=0}^{\infty}$, can be chosen adaptively, based on the difficulty of minimizing $Q(\mathbf{w}_k, \rho_k)$ at the $k^{th}$ iteration. The sequence of solutions, $\{\mathbf{w}_k\}_{k=0}^{\infty}$, converges to the optimal solution $x^*$ as the sequence of penalty parameters, $\{\rho_k\}_{k=0}^{\infty}$, tends to $\infty$. However, the minimization of $Q(\mathbf{w}_k, \rho_k)$ becomes more and more difficult when $\rho_k$ becomes large [80].

A third approach to solving the problem in (3.1.5) is the so-called the *augmented Lagrangian function method* given by

$$L_A(\mathbf{w}, \lambda, \rho) = f(\mathbf{w}) - \sum_{i \in \xi} \lambda_i c_i(\mathbf{w}) + \frac{\rho}{2} \sum_{i \in \xi} c_i^2(\mathbf{w}). \tag{3.1.13}$$

A sequence of minimizations of the form

$$\min_{\mathbf{w} \in \Re^n} L_A(\mathbf{w}, \lambda_k, \rho_k) \tag{3.1.14}$$

is performed. Note that (3.1.14) is in the form of an unconstrained minimization problem in $x$ alone. The initial $\lambda_0$ and $\rho_0$ are chosen a priori. The parameter sequence,$\{\lambda_k\}_{k=0}^{\infty}$ and $\{\rho_k\}_{k=0}^{\infty}$, are updated in a general algorithm to ensure global convergence. This can be achieved by Powell's method given below [27].

---

Powell's method

---

(i) Initially set $\lambda = \lambda^{(0)}$, $\rho = \rho^{(0)}$, $k = 0$, $||c^{(0)}||_\infty = \infty$.

(ii) Find the minimizer of $L_A$ and denote $c = c(\mathbf{w}, \lambda, \rho)$.

(iii) If $||c||_\infty > \frac{1}{4}||c^{(k)}||_\infty$, set $\rho = 10\rho$ and go to (ii).

(iv) Set $k = k + 1$, $\lambda^{(k)} = \lambda$, $\rho^{(k)} = \rho$, $c^{(k)} = c$.

(v) Set $\lambda = \lambda^{(k)} - \rho^{(k)}c^{(k)}$ and go to (ii).

---

It turns out that by combining features of the (ordinary) Lagrangian function and the penalty methods, the augmented Lagrangian function method moderates the disadvantages of both. Convergence in the augmented Lagrangian method can usually be attained without the need to increase $\rho_k$ to $\infty$. In addition, the augmented Lagrangian iterations (3.1.13) tend to converge to a Lagrangian multiplier much faster than the (ordinary) Lagrangian iterations(3.1.11).

The problem with inequality constraints in (3.1.6) can be converted to a problem with equality constraints by introducing *slack variables* in the augmented Lagrangian function as follows:

$$c_i(\mathbf{w}) - s_i = 0, s_i \geq 0, i \in \Im.$$

After eliminating these slack variables by an explicit minimization with respect to

each $s_i$, we obtain an equivalent form of the subproblem (3.1.14) as

$$\min \ \{f(\mathbf{w}) + \sum_{i \in \Im} \psi(\mathbf{w}_k, \lambda_i^k, \rho^k)\}, \tag{3.1.16}$$

where

$$\psi(\mathbf{w}_k, \lambda_i^k, \rho^k) = \begin{cases} -\lambda_i^k c_i(\mathbf{w}_k) + \frac{\rho^k}{2} c_i^2(\mathbf{w}_k) & , \text{ if } c_i(\mathbf{w}_k) - \frac{\lambda_i^k}{\rho_k} \leq 0 \\ -\frac{(\lambda_i^k)^2}{2\rho^k} & , \text{ otherwise.} \end{cases} \tag{3.1.17}$$

## *3.2 Multi-Objective Programming*

Suppose that we are given a set of $K$ design objectives, $f_k(\Theta) \in (0,1)$ , $k = 1, ..., K$, each of which is a nonlinear function of $\Theta$. The best decision vectors, $\hat{\Theta} = \{\hat{\mathbf{w}}_1, ..., \hat{\mathbf{w}}_M\}$, are found by MOP, which is formulated as

$$\min_{\Theta} \quad [f_1(\Theta), f_2(\Theta), ..., f_K(\Theta)], \tag{3.2.1}$$

$$\text{subject to} \quad \Theta \in \{g(\Theta) = [g_1(\Theta), g_2(\Theta), ..., g_m(\Theta)]^T \leq 0\}. \tag{3.2.2}$$

For clarity we assume that all the objective functions are to be minimized. If an objective function $f_i$ is to be maximized, it is equivalent to minimizing the function $-f_i$. The word "minimize" means that we want to minimize all the objective functions simultaneously. If a multi objective problem is well formed, there does not exist a single solution that simultaneously minimizes each objective to its fullest. This means that the objective functions are at least partially conflicting. MOP looks for a solution for which each objective has been optimized to the extent that improvement any one of the objectives would degrade the others. The goals of MOP are to find such a solution and to quantify how much better this solution is compared to other such solutions.

### 3.2.1 Pareto Optimality

In SOP problems, we say that a solution with a smaller objective function value is better than another solution with a large objective function value. However, there

is no such natural ordering in the objective space for multi-objective problems. For example, let $(f_1, f_2)$ denote two objective function values. It can be said that $(1, 1)$ is less than $(3, 3)$, but it is not obvious how to compare $(1, 3)$ and $(3, 1)$.

**Definition 3.2.1** *A decision vector $\Theta^*$ is (global) Pareto optimal if there does not exist another decision vector $\Theta$ such that $f_i(\Theta) \leq f_i(\Theta^*), \forall i = 1, ..., M$ and $f_j(\Theta) < f_j(\Theta^*)$ at least for one index $j$.*

Several other terms such as *non-inferiority*, *efficiency* and *non-dominance* are used for the Pareto optimality concept defined above. There are usually (infinitely) many Pareto optimal solutions. Hence, we speak of a *Pareto optimal set*. In Figure 5, a feasible objective region $Z \in \Re^M$ is illustrated. The thick line contains the set of all Pareto optimal vectors. Mathematically, every Pareto optimal solution is an equally acceptable solution to the MOP problem. According to the definition of Pareto optimality, moving from one Pareto optimal solution to the other necessitates trading off. Usually, a decision maker is included in the problem to select one out of the set of Pareto optimal solutions.

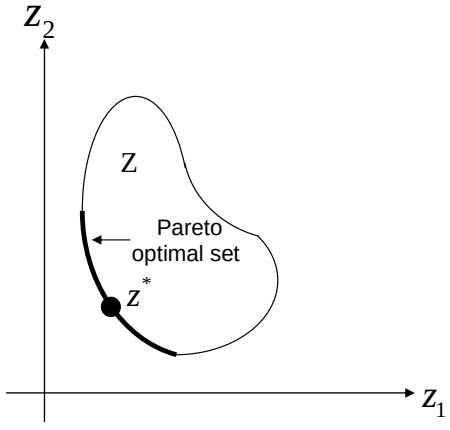



**Figure 5:** An illustration of Pareto optimal solutions in a multi-objective problem.

*Trade-offs* and *marginal rates of substitution* are related to changes in the objective function values when we move from one solution to another. A tradeoff reflects the

ratio of change in the values of the objective functions when the value of an objective function decreases.

**Definition 3.2.2** *Let $x^1, x^2 \in \chi$ be two decision vectors with corresponding objective function vectors, $f(\Theta^1)$ and $f(\Theta^2)$, respectively. We denote the ratio of change between the functions $f_i$ and $f_j$ by*

$$\Lambda_{ij} = \Lambda_{ij}(\Theta^1, \Theta^2) = \frac{f_i(\Theta^1) - f_i(\Theta^2)}{f_j(\Theta^1) - f_j(\Theta^2)}, \tag{3.2.3}$$

*where $f_j(\Theta^1) - f_j(\Theta^2) \neq 0$. $\Lambda_{ij}$ is called the partial tradeoff involving $f_i$ and $f_j$ between $\Theta^1$ and $\Theta^2$ [102].*

It is said that two feasible solutions are situated on the same *indifference curve* if the decision maker finds them equally desirable, as illustrated in Figure 6. The final Pareto optimal solution is situated on an indifference curve where it is tangent to the Pareto optimal set.

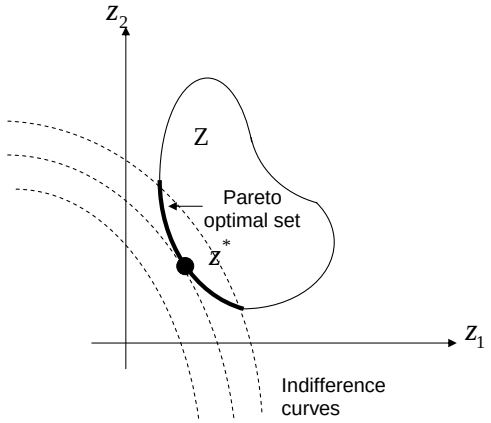



**Figure 6:** The Pareto optimal set and indifference curves.

#### *3.2.1.1 First-Order Pareto Optimality Conditions*

The optimality conditions for MOP problems have been investigated by many researchers such as Kuhn and Tucker [53], Da Cunha and Polak [21], Marisciac [68].

**Theorem 3.2.1** *(Fritz Jogn necessary condition for Pareto optimality) Let the objective and constraint functions of the MOP problem be continuously differentiable at* $\Theta^*$. *A necessary condition for* $\Theta^*$ *to be Pareto optimal is that there exists vectors* $0 \leq \lambda \in \Re^k$ *and* $0 \leq \mu \in \Re^m$ *for which* $(\lambda, \mu) \neq (0, 0)$ *such that [21]*

$$(1) \qquad \sum_{i=1}^{k} \lambda_i \nabla f_i(\Theta^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(\Theta^*) = 0$$

$$(2) \qquad \mu_j g_j(\Theta^*) = 0 \text{ for all } j = 1, ..., m.$$

This condition is also a sufficient condition for $x^*$ to be Pareto optimal if the objective and the constraint functions are convex and continuously differentiable at $\Theta^*$.

In addition to the first order optimality conditions, second-order optimality conditions provide a means of reducing the set of candidate solutions produced by the first-order conditions. However, second-order optimality conditions have been examined substantially less than first-order optimality conditions. See [107] for a detailed discussion of the second-order Pareto optimality conditions.

### 3.2.2 MOP Methods

Methods of MOP can be classified according to the participation of the decision maker in the solution process as follows:

1. *No-preference methods*, where no articulation of preference information is used
2. *A posteriori methods*, where a posteriori articulation of preference information is used
3. *A priori methods*, where a priori articulation of preference information is used
4. *Interactive methods*, where progressive articulation of preference information is used.

In the following, several methods are presented. None of them can be said to be superior to others. When selecting a method, the specific features of the problem should be taken into consideration. The general selecting criteria are appropriateness, ease of use, validity, and sensitivity of the results to the choice of the method.

#### *3.2.2.1 No-Preference Methods*

In the no-preference methods, the multi-objective problem is solved using some relatively simple method, and the solution obtained is presented to the decision maker. These methods are suitable for situations where the decision maker does not have any special expectation.

*The Method of Global Criterion:*

The method of global criterion is sometimes called *compromise programming*. In this method, the distance between a reference point and the feasible objective region is minimized. The analyst has to select a reference point $\bar{\mathbf{z}}$ and a metric for measuring the distances. One possible approach is using the $L_p$-metric, which corresponds to solving

$$\min_{\Theta} \left( \sum_{i=1}^{M} |f_i(\Theta) - \bar{z}_i|^p \right)^{1/p}.$$

#### *3.2.2.2 A Posteriori Methods*

In a posteriori methods, after the Pareto optimal set is generated, it is presented to the decision maker, who selects the most preferred solution among the alternatives. In general, the generation of the Pareto optimal set is computationally expensive.

*Weighting Method:*

The idea in the weighting method is to minimize a weighted sum of the objectives. In this way, the multi-objective problem is converted into

$$\min_{\Theta} \sum_{i=1}^{M} \alpha_i f_i(\Theta),$$

where $\alpha_i \geq 0, \forall i = 1, ..., M$, and $\sum_{i=1}^{M} \alpha_i = 1$.

As an extension to the global criterion method, the $L_p$ metrics can also be weighted in order to produce different (weakly) Pareto optimal points.

$$\min_{\Theta} \left( \sum_{i=1}^{M} \alpha_i |f_i(\Theta) - \bar{z}_i|^p \right)^{1/p}$$

#### *3.2.2.3 A Priori Methods*

In the case of a priori methods, the decision maker must specify his or her preference, hopes, and opinions at the beginning. The difficulty is that the decision maker does not necessarily know beforehand what is possible to attain, and how realistic his or her expectations are.

*Preference Function Method:*

In the value function method, the decision maker should be able to give an accurate and explicit mathematical form of a *preference function*, $U : \Re^k \rightarrow \Re$, that represents his or her preference globally. Then, the preference function problem

$$\max_{\Theta} U(f(\Theta))$$

can be solved by an SOP method. The preference function method could be called the optimal way of solving multi-objective optimization problems if the decision maker could reliably express the value function. The difficulty with the preference function method lies in specifying the mathematical expression of the preference function. Furthermore, even if it were possible for the decision maker to express his or her preferences globally, the resulting preference structure might be too simple, since the preference functions cannot represent intransitivity or incomparatibility.

*Lexicographic Ordering:*

In lexicographic ordering method, the decision maker must arrange the objective functions according to their absolute importance. After ordering, the most important

objective function is minimized subject to the original constraints. If this problem has a unique solution, it is the solution of the original multi-objective optimization problem. Otherwise, the second most important objective function is minimized with the additional constraint for the most important objective to guarantee that most important objective function preserves its optimal value.

The solution to the lexicographic ordering method is Pareto optimal. However, it has several drawbacks. The decision maker may have difficulties in putting the objective functions into an absolute order of importance. It is also very likely that the less important objective functions are not taken into consideration at all.

*Goal Programming:*

The idea is to establish a hierarchy of importance among the incompatible goals by minimizing the deviations from (optimistic) aspiration levels set for each goal. An objective function jointly with an aspiration level forms a *goal*. Most goal programming algorithms have been developed in the linear framework [13], but many techniques are also applicable to nonlinear problems. Saber and Ravindran [91] give a thorough review of nonlinear goal programming. Goal programming is a very widely used solution for practical MOP problems. However, it is not an appropriate method to use if it is desired to obtain tradeoffs.

#### *3.2.2.4 Interactive Methods*

In interactive methods, the decision maker works together with an analyst or an interactive computer program. A solution pattern is formed and repeated several times. After every iteration, some information is given to the decision maker. Assuming that the decision maker has enough time and capabilities for co-operation, part of the Pareto optimal points are generated and evaluated. The decision maker can specify and correct his/her preferences and selections as the solution process continues. Although the decision maker does not need to know any global preference structure,

the interactive methods have several difficulties. Consistency of the responses from the decision maker is one of the most important factors guaranteeing the success of interactive solution methods. Different starting points, different types of questions or interaction styles may lead to different final solutions [71, 72].

# CHAPTER IV

# LITERATURE SURVEY

This chapter is devoted to the related work on the two applications that we consider in this dissertation, namely, on statistical language modeling and adaptation and automatic spoken language identification. The last part reviews some of the most well-known MOP-based frameworks developed for related applications.

## *4.1 Literature Survey on Language Modeling and Adaptation*

The last few decades have witnessed significant progress toward increasing the predictive capacity of statistical models of human language. These statistical models find applications in speech recognition, machine translation, part-of-speech tagging, parsing, information retrieval, and optical character recognition. This chapter is about the use of language modeling for automatic speech recognition. Speech recognition is concerned with the process of converting an acoustic signal containing speech data into the appropriate text transcription. Human-like speech recognition performance cannot be achieved by considering the acoustic signal alone and some form of linguistic knowledge is essential. Moreover, language modeling is not merely useful in order to improve the performance of a speech recognition system. It is necessary in order to be able to distinguish between homonyms such as "to" and "two", a distinction which would be impossible using only the information contained within the acoustic signal.

Two major approaches to the modeling of human language may be identified. The first relies on syntactic and semantic analysis of the sample text to determine the hierarchical sentence structure. Such analysis employs a set of rules to ascertain

whether a sentence is permissible or not. Although it has been possible to describe a significant proportion of language usage in this way, complete coverage has remained elusive. This comes from the continuous changes taking place in a living language. Furthermore, utterances that are clearly not grammatical occur often in a natural language, but cannot be dealt with by such analysis. This significant likelihood of failure under such naturally occurring circumstances has led to infrequent use of the rule-based approach in many systems.

Instead, a second approach based on statistical techniques with intrinsically greater robustness to grammatical irregularities is usually taken. Such statistical language models assign to each word in an utterance a probability value according to its deemed likelihood within the context of the surrounding word sequence. The probabilities are inferred from a large body of example text, referred to as the training corpus. In this way the model may reflect language usage as found in practice, and not only its grammatical idealization. Moreover, advantage may be taken of the recent vast increases in the amount of available training text, which now runs into hundreds of millions of words.

Perhaps the major problem with such statistical systems is the limitations in the data that are used to estimate the system behavior. In general, the more data we have, the more reliable our model will be. However, collecting and processing huge amounts of data is often unrealizable. Therefore, the goal in applications is to use the limited amount of data intelligently to develop highly reliable statistical models. For instance, in the natural language grammar problem, if we are given only a small amount of data that we cannot truly rely on to infer a model from, we can use some other data which is somewhat related. As an example, we might be interested in inferring a model to estimate the probabilities of sequences of words in weather-forecasting news for which we have a limited amount of data. In addition, we may have a huge amount of data collected for estimating such probabilities in everyday

**Table 1:** The statistics for the WSJ0 data set.

| | $\mathbb{O}$ | $\mathbb{T}$ | | |
|---|---|---|---|---|
| | Total $n$-grams | Total $n$-grams | Seen in $\mathbb{O}$ | Unseen in $\mathbb{O}$ |
| $c_1$ | 4,985 | 1,272 | 1,272 (100%) | - |
| $c_2$ | 1,639,687 | 3,438 | 3,301 (96%) | 137 (4%) |
| $c_3$ | 9,181,362 | 4,272 | 3,418 (80%) | 854 (20%) |

news, which contains weather-forecasting news as well. In such cases, we can merge information estimated from these datasets. Doing so requires a decision though: How faithful we should be to the general domain dataset and to the application-specific dataset in developing a model.

The language modeling typically requires millions of probabilities to be estimated. In a sparse sample, however, the maximum likelihood estimate given in (1.4.6) is biased high for observed events and biased low for unobserved ones. In an $n$-gram LM it is impossible to avoid the problem of unseen events. For instance, the Wall Street Journal (WSJ0) data set [81] is composed of more than 1.6 million standardized sentences collected from 1987 until 1989, and the test domain data is composed of newswire stories collected in November 1992. F,

Table 1 shows the number of $n$-grams, $c_n$, for the commonly performed 5K-word vocabulary ASR task. There are 25 million possible bigrams and 125 billion possible trigrams, whose probabilities should be calculated from the available training set, $\mathbb{O}$. Not only is a very small portion of possible $n$-grams ever observed in the training set, 4% of the test bigrams and 20% of the test trigrams are never observed in $\mathbb{O}$.

### 4.1.1 The Limitations of $n$-gram LMs

The choice of $n$ is based on a trade-off between detail and reliability, and is dependent on the quantity of training data available. A bigram (i.e., $n = 2$) LM will have larger equivalence classes, and hence fewer parameters than a trigram (i.e., $n = 3$) LM. A bigram LM's parameters are therefore more reliably estimated, while a trigram LM

is more detailed, and therefore is a more accurate model, provided there is sufficient training data. Typically, trigram LMs strike the best balance between detail and reliability, although interest is growing in moving to 4-gram models and beyond.

The simplicity of $n$-gram models is both their greatest strength and weakness. On the positive side, the models' simplicity means that the models are easy and efficient to train, even when corpora of hundreds of millions of words are used. They are also very powerful, and surprisingly difficult to improve on [43]. Unfortunately, they are also seriously deficient:

1. The key difficulty with $n$-gram LMs is that of *data sparsity*. One can never have enough training data to estimate all of the model's parameters reliably.

2. They are completely "blind" to any phenomenon, or constraint, that is outside their limited scope. As a result, nonsensical and even ungrammatical utterances may receive high scores as long as they do not violate local constraints.

3. The predictors in $n$-gram models are defined by their ordinal place in the sentence, not by their linguistic role. The word sequences "*gold prices fell yesterday to*" and "*gold prices fell to*" seem different to a trigram LM, yet they are likely to have a similar effect on the distribution of the next word.

### 4.1.2 Class-Based $n$-gram LMs

The parameter space spanned by $n$-gram LMs can be significantly reduced, and reliability of estimates consequently increased, by clustering the words into classes. This can be done at many different levels: one or more of the predictors may be clustered or the predicted word itself may be associated with a class label. For instance, the city names "Atlanta", "Los Angeles", and etc. can be grouped into a generic class "CITY". With a class-based $n$-gram LM, we would have

$$P(\omega_n|\omega_1, ..., \omega_{n-1}) = P(c_n|c_1, ..., c_{n-1})P(\omega_n|c_n) \tag{4.1.1}$$

where $c_i$ is the class that $\omega_i$ belongs to. Taking this approach has the following advantages:

- Class-based models share statistics between words of the same category and are therefore able to generalize to word patterns never encountered in the training corpus.
- Grouping words into classes can reduce the number of contexts in a model and thereby alleviate data sparsity problem.

The decision as to which components to cluster, as well as the nature and extent of the clustering, are other examples of the detail versus reliability tradeoff. In addition, one must decide on the clustering itself. There are three general methods for doing so: (i) Clustering by linguistic knowledge, (ii) Clustering by Domain knowledge, (iii) Data driven clustering.

### 4.1.3 Long-Distance $n$-gram LMs

Long-distance $n$-gram LMs attempt to capture directly the dependence of the predicted word on $(n-1)$grams which are some distance back. For example, a distance-2 trigram predicts $\omega_i$ based on $(\omega_{i-3}, \omega_{i-2})$. As a special case, distance-1 $n$-grams are the familiar conventional $n$-grams. Although they capture word-sequence correlations even when the sequences are separated by a distance $d$, they fail to appropriately merge training instances that are based on different values of $d$.

### 4.1.4 LM Smoothing

The term *smoothing* describes techniques for making the distributions more uniform by adjusting low probabilities such as zero probabilities upward and high probabilities downward. Not only do smoothing methods generally prevent zero probabilities, but they also attempt to improve the accuracy of the model as a whole.

One of the simplest types of smoothing used in practice is additive smoothing [46, 62], in which we pretend each $n$-gram occurs $\delta$ times more than it actually does, where typically $0 < \delta \leq 1$, i.e.,

$$p(\omega_i|\omega_{i-n+1}^{i-1}) = \frac{\delta + c(\omega_{i-n+1}^{i})}{\delta|V| + \sum_{\omega_i} c(\omega_{i-n+1}^{i})}.$$

The Good-Turing estimate [33] is central to many smoothing techniques. The Good-Turing estimate states that for any $n$-gram that occurs $r$ times, we should pretend that it occurs $r^*$ times where

$$r^* = (r+1)\frac{n_{r+1}}{n_r}$$

, where $n_r$ is the number of $n$-grams that occur exactly $r$ times in the training data. However, doing this gives too much of the probability space to unseen events.

As another smoothing technique, it is useful to interpolate higher order $n$-gram models with lower order $n$-gram models. When there is insufficient data to estimate a probability in the higher order model, the lower-order model can often provide useful information. A general class of interpolated models is given as follows [12]:

$$P_{interp}(\omega_i|\omega_{i-n+1}^{i-1}) = \lambda_{i-n+1}^{i-1} p_{ML}(\omega_i|\omega_{i-n+1}^{i-1}) + (1 - \lambda_{i-n+1}^{i-1}) P_{interp}(\omega_i|\omega_{i-n+2}^{i-1}).$$

That is, the $n^{th}$ order smoothed model is defined recursively as a linear interpolation between the $n^{th}$ order maximum likelihood model and the $(n-1)^{th}$ order smoothed model. To end the recursion, we can take the smoothed $1^{st}$ order model to be the maximum likelihood distribution, or we can take the smoothed $0^{th}$ order model to be the uniform distribution.

### 4.1.5 LM Back-off

In the back-off method [48], the different information sources are ranked in order of detail. At runtime, if the most detailed model is found to contain enough information

about the predicted word in the current context, then that context is used exclusively to generate the estimate. Otherwise, the next model in line is consulted. Back-off can be used both as a way of combining information sources and as a way of smoothing. The back-off method does not actually reconcile multiple models. Instead, it chooses among them.

Katz smoothing extends the intuitions of the Good-Turing estimate by adding the combination of higher-order models with lower-order models. For a bigram $w_{i-1}^{i}$ with count $r = c(w_{i-1}^{i})$, its corrected count, $r^*$, is calculated using the equation

$$r^* = c_{Katz}(w_{i-1}^{i}) = \begin{cases} d_r r & , \text{ if } r > 0 \\ \alpha(\omega_{i-n+1}^{i-1}) P_{ML}(\omega_i | \omega_{i-n+2}^{i-1}) & , \text{ if } r = 0 \end{cases}$$

That is, all bigrams with a nonzero count $r$ are discounted according to a discount ratio, $d_r$, which is approximately $r^*/r$ and given as

$$d_r = \begin{cases} \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} & , \text{ if } rleqk \\ \alpha(\omega_{i-n+1}^{i-1}) P_{ML}(\omega_i | \omega_{i-n+2}^{i-1}) & , \text{ if } r > k \end{cases}$$

where Katz suggests $k = 5$ and the backoff weight $\alpha(\omega_{i-n+1}^{i-1})$ is given as

$$\alpha(\omega_{i-n+1}^{i-1}) = \frac{1 - \sum_{\omega_i : c(\omega_{i-n+1}^{i}) > 0} P_{Katz}(\omega_i | \omega_{i-1})}{1 - \sum_{\omega_i : c(\omega_{i-n+1}^{i}) > 0} P_{ML}(\omega_i)}. \tag{4.1.7}$$

One problem with this approach is that it exhibits a discontinuity around the point where the back-off decision is made. In spite of this problem, backing off is simple and compact. Another problem with back-off is that it gives rise to systematic overestimation of some events [89].

### 4.1.6 An Alternative: Stochastic Decision Tree-Based LMs

In [4], an LM training technique is presented that adopts a radically different approach from the $n$-gram model. The model is based on a binary decision tree. At each node of the tree, a "YES/NO" question about the word history is asked, e.g., "Was the

previous word a verb?", "Was there any computer jargon in the previous fifty words?". The answer to these questions determines the path taken down the tree from the root node to a leaf. Thus the word histories are partitioned into equivalence classes by the decision tree, instead of the most recent $n-1$ words. At each leaf node, a probability distribution is defined over all the words in the vocabulary. The model that is proposed achieves a moderate improvement in perplexity over a trigram model trained on the same data.

The key difficulty with this approach is that of training a good tree from the training data. The list of potential questions one could ask at each node is so vast that searching through the space of potential trees is computationally expensive. Clearly, some reduction of the set of possible questions is necessary, and it is in this simplification that the tree-based models lose much of their power.

Although there can be other alternatives to $n$-gram models, none outperforms $n$-gram models regardless of however complicated they are. For these reasons, there has been en effort to *adapt* $n$-gram LMs to specific application domains instead of totally avoiding $n$-grams.

### 4.1.7 LM Adaptation

LMs trained on large quantities of text covering many subject areas and styles of writing display good average performance, but they are not able to take advantage of the particularities of the domains to which they are applied. Moreover, often there are insufficient data from the specific application domain to allow specialized models to be built directly. *Adaptivity* in an LM concerns its capacity to alter the probability estimate, $P(W)$, in accordance with the particular nature of the text. Such text domains may be distinguished by attributes such as the topic of discussion, style of writing, and the time of writing [88]. There have been a variety of LM adaptation techniques proposed, the most successful of which are briefly reviewed

here (see e.g., [4, 5, 15, 85, 108] for further techniques).

#### *4.1.7.1 Linear Interpolation*

Given $K$ LMs, $P_k$, which are possibly estimated from different datasets, we can combine them linearly with weights typically found with an Estimation-Maximization (EM) type algorithm [23]. The *target LM* (i.e., the adapted LM) is then

$$P(\omega|h_\omega) = \sum_{k=1}^{K} \nu_k P_k(\omega|h_\omega), \tag{4.1.8}$$

where $\nu_k \geq 0, \sum_k \nu_k = 1$. Linear interpolation is easy to implement and is guaranteed to be no worse than any of its components. This is because each of the component LMs can be viewed as a special case of the interpolation, with a weight of 1 for that component and 0 for all others. However, linearly interpolated models make suboptimal use of their components. The different information sources are consulted blindly, without regard to their strengths and weaknesses in particular contexts. Moreover, their weights are optimized globally, not locally (i.e., not for individual histories or words).

#### *4.1.7.2 Log-Linear Interpolation*

A related approach was taken in [51]. Suppose a set of LM models that are to be merged $P_k$ are given. We wish to minimize the distance to a well-trained model, $P_0$. At the same time the constraints that the KL divergence of the unknown model to the given models $P_k$ is $D(P||P_k) = d_k$ should be satisfied ($d_k$ are not known a priori). Then, the target LM probabilities, $P(\omega|h)$, minimize the Lagrangian function

$$L = D(p||p_0) + \sum_k \lambda_k (D(P||P_k) - d_k). \tag{4.1.9}$$

Solving this problem, we obtain,

$$P(\omega|h) = \frac{1}{Z_\lambda(h)} \prod_k P_k(\omega|h)^{\lambda_k}. \tag{4.1.10}$$

There are two sets of free parameters, namely $d_k$ or $\lambda_k$. The LM weights, $\lambda_k$, can be found by optimizing the log-likelihood in a cross-validation set.

#### *4.1.7.3 MAP LM Adaptation*

MAP-based approaches [3, 25, 69] use a prior distribution to exploit how much the $n$-gram estimates in the specific application domain diverge from the background estimates. In [25], a Dirichlet distribution is used to represent the prior distribution of $n$-grams. This gives the following MAP probabilities for $n$-grams:

$$P_{MAP}(\omega|h) = \left(\frac{|S|}{S+\lambda A}\right) P_{ML}(\omega|h) + \left(\frac{\lambda|A|}{S+\lambda A}\right) P_{ML}(\omega|h).$$

In [69], a beta distribution is used as the prior distribution, which results in the following $n$-gram probabilities:

$$P_{MAP}(\omega|h) = \frac{c_A(h,\omega) + \alpha - a}{\sum_{\omega^p rime} c_A(h,\omega') + \alpha + \beta - 2}$$

where $\alpha$ and $\beta$ are the hyper-parameters of the beta distribution. In [3], the MAP probabilities are estimated as

$$P_{MAP}(\omega|h) = \frac{\lambda_h c_S(h,\omega) + c_A(h,\omega)}{\lambda_h \sum_{\omega'} c_S(h,\omega') + \sum_{\omega^p rime} c_A(h,\omega')}$$

where $\lambda_h$ is a history-dependent parameter that indicates how much the prior counts should be relied upon. All these estimation methods have improved the performance of the model to some extend.

#### *4.1.7.4 Other Approaches to LM Adaptation*

*Cache LMs*:

LMs usually employ probability estimates that have been chosen to perform well on average over the entire training corpus. This precludes adaptation to dynamic changes in the text characteristics, and therefore such models are described as static.

The underlying philosophy of a cache is that, due to local text characteristics such as topic and author, words or word patterns that have occurred recently are more likely to recur in the immediate future than a static language model would predict. A cache consists of a buffer of the most recent words of text from which LM probabilities are calculated [54, 55, 86]. The cache LM probabilities, $P_{cache}$, are combined with the static model probabilities, $P$, by linear interpolation.

*Mixtures of Topic-Specific LMs*:

In order to account for a number of distinct themes appearing in a corpus, the text may be divided into partitions corresponding to common subject matter, termed topics, following which a trigram language model is built for each individual topic [14,40,93]. The resulting models are then combined linearly to obtain an overall probability estimate.

*Constraint-Specification Approaches*:

Constraint specification approaches [83] use the application-specific data set to extract features that the adapted LM is constrained to satisfy. Typically, a target LM is trained by maximizing its entropy with all these constraints satisfied.

### 4.1.8 LM Corpora

In the LM adaptation problem, there are three different data sets. The first one is a very large general domain dataset, denoted as $\mathbb{S}$. This dataset is not specific to the application, but provides some "background" or "prior" information about the application. The second dataset is the test set that we attempt to develop the most suitable model for. The third dataset is the application-specific dataset or the adaptation set, denoted as $\mathbb{A}$, which is closely related to the test dataset, but unfortunately not large enough to derive reliable models from. The LM adaptation techniques differ in the way they use these two datasets to obtain the target model [8, 87].

In the LM adaptation experiments, we use the ARPA WSJ language corpus. The first ARPA CSR Wall Street Journal corpus consists of articles published in the Wall Street Journal from December 1986 through November 1989. The original data of more than 1.6 million standardized sentences was obtained, conditioned, and processed for linguistic research by the Association for Computational Linguistics' Data Collection Initiative (ACL/DCI). The corpus was chosen by the ARPA speech recognition community to be the basis for its Continuous Speech Recognition (CSR) common evaluation project.

Subsequently, most of the data was further processed at MIT Lincoln Labs [82], and conditioned for use in speech recognition. This included transforming many common text constructs to the way they are likely to be said when read aloud (e.g., "123.45 dollars" might be transformed into "A hundred and twenty three dollars and forty five cents"), some quality filtering, preparation of various standard vocabularies, and much more. We refer to this data set as the WSJ corpus. The version of this corpus used in the experiments described in this paper is the one where punctuation marks were assumed not to be verbalized, and were thus removed from the data. This was known as the non-verbalized-punctuation (nvp) condition. The pseudo word "$< /s >$" was added to the vocabulary to designate end-of-sentence. The pseudo word "$< s >$" was used to designate beginning-of-sentence, but was not made part of the vocabulary.

## *4.2 Literature Survey on LID*

Automatic language identification (LID) is the problem of identifying the language being spoken by an unknown speaker from a sample of speech. LID is often used as a front-end system to a language-specific speech recognition system for applications such as directory assistance, machine translation, and multi-lingual information retrieval. As with speech recognition, humans are the most accurate LID systems:

Within seconds of hearing speech, people are able to determine whether it is a language they know. If it is a language with which they are not familiar, they can often make subjective judgments as to its similarity to a language they know, e.g., they can say "It sounds like German".

The key to solving the problem of LID is the detection and exploitation of differences between languages. It is desirable to discover *language discriminating characteristics* that are relatively easy to extract from the acoustic signal, do not require complex methodologies to model, and are relatively free of noise from speaker, channel, and vocabulary dependencies. Languages have characteristic sound patterns and differ in the inventory of phonological units (speech sound categories) used to produce words, the frequency of the occurrences of these units, and the order in which they occur in words. Aside from these, there may also be significant differences in the acoustic realizations of particular phones across languages. Prosodic patterns also differ significantly between languages. A variety of approaches have been developed to solve the LID problem, which differ in the information sources being used.

### 4.2.1 Information Sources for LID

It may be possible to develop an LID system based only on the information that is directly available from the waveform of a spoken utterance. The information that is available in an utterance's waveform can be viewed as belonging to one of two groups: phonological information and prosodic information. Furthermore, long segments of speech carry linguistic information, which truly distinguish languages from each other.

#### *4.2.1.1 Phonological Information*

The phonological properties of a spoken utterance can vary greatly from language to language. There are various phonological factors, which help define the distinctiveness of a language. Some of these factors include the phone set, the phonotactic constraints, and the acoustic realizations of particular phones within a language.

Because each language uses only a small subset of phones from the set of all possible speech sounds, variances can be observed across the phone sets of different languages. Thus, knowledge of the phones used in particular languages may be enough to help distinguish one language from another. Even if languages contain nearly identical phone sets, the languages may still be distinguishable by the probability distribution of the phones across each language.

Significant differences may also exist in the acoustic realizations of particular phones across different languages. These differences may be caused by cross-language differences in the articulatory gestures used to produce the phone. For example, the phoneme "/t/" can be realized by a large set of allophones. It can be realized with or without aspiration, with a dental or alveolar closure, and with lips rounded or unrounded. The use of each of these allophones varies across languages.

Some differences in the acoustic realizations of particular phones across languages may occur because of the particular phonotactic constraints present within each language. The phonotactic constraints of different languages may cause certain coarticulation effects to be possible in one language but not possible in another.

#### *4.2.1.2 Prosodic Information*

The prosodic properties of languages can also vary greatly. Fundamental frequency, duration, and voice intensity are all important elements used within the prosodic structure of a spoken utterance. The manner in which these elements are incorporated into the prosodic structure of an utterance varies across languages. The differences across languages can often be observed in the realization of the prosodic features, which determine the tones or stress contained throughout an utterance.

In languages that incorporate the concept of word stress, intensity, duration, and fundamental frequency contor of a syllable are all correlated with the inherent stress

being placed on that particular syllable [103]. Different languages use stress in different manners. For free stress languages, such as English, the stress pattern of two words with the same number of syllables can be different. However, for fixed stress languages, such as Polish, the stress pattern is dependent only on the number of syllables present in each word [90].

The effect of pre-pausal lengthening of vowels is another prosodic effect which has been observed to differ across languages. Lengthening of the final vowel in a sentence is a readily observable characteristic of spoken utterances in English, French, German, and Italian. However, other languages such as Finnish, Estonian, and Japanese have been observed to contain little or no sentence-final lengthening of vowels [103].

#### *4.2.1.3 Phonotactic Information*

Different languages also have different rules governing how sequences of phonemes may be constructed to form higher level linguistic elements such as syllables or words. *Phonotactics* refers to the study of the constraints on relative frequencies of sound units and their sequences in speech. Its popularity in LID is due to its relatively high language-discriminating power. The phonotactic constraints cause certain phonetic sequences to be likely in some languages but unlikely in others. For example, Japanese has strict phonotactic constraints which generally prohibit consonants from following consonants. English, on the other hand, has looser constraints, which allow for the possibility of multiple consonants in succession.

### 4.2.2 Approaches to LID

It is possible to distinguish four categories of approaches which differ in the information sources being used. These four categories are

1. Acoustic approaches,

2. Phonotactic approaches,

3. Approaches that use prosodic and duration information,

4. Discriminative approaches.

#### *4.2.2.1 Acoustic Approaches to LID*

Purely acoustic LID aims at capturing the essential differences between languages by modeling distributions of spectral features directly. This is typically done by extracting a language-independent set of spectral features from speech and using a statistical classifier to identify the language-dependent patterns in such features. The classification of a given acoustic signal $x$ into a language $L^*$ is then given by

$$L^* = \arg\max_L P(L|x) = \arg\max_L P(x|L)P(L) \tag{4.2.1}$$

where $P(L)$ and $P(L|x)$ are the a priori and the a posteriori probabilities of the language $L$, respectively, and $P(x|L)$ is the acoustic probability of $x$ for the language $L$. Acoustic approaches were among the earliest studies of LID. In [18], an LID system for eight languages was developed on the basis of linear predictive analysis. In [28], prosodic features (energy, zero-crossing rate, and their derivatives), and formant positions were used to train vector quantization (VQ) codebooks for each language.

Gaussian mixture models (GMMs) have successfully been applied to modeling the acoustic probability, $P(x|L)$, for LID [100, 113]. The GMM LID system consists of a feature extraction pre-processor, a GMM for each target language, and a back-end classifier. GMMs trained on speech data from the target languages produce acoustic class conditional likelihood scores for each test utterance.

All acoustic modeling approaches presented so far model only the static distribution of acoustic features given the language. However, much language-discriminating information resides in the dynamic patterns of acoustic features changing over time. Since there is no intermediate mapping to explicit linguistic units (such as phones or syllables), differences in acoustic features may represent a variety of phenomena such

as pronunciations of individual sounds across languages or co-articulation of sounds in sequence.

#### *4.2.2.2 Phonotactic Approaches to LID*

State-of-the-art LID systems utilize complete knowledge of lexical and grammatical structure of a language. They typically use fully fledged large vocabulary continuous speech recognizers (LVCSRs) to decode an incoming utterance into strings of words with a subsequent analysis for LID-specific patterns. The phonotactic LID method uses a probabilistic framework and builds on the inherent property of every language to exhibit strong language-specific frequencies and dependencies between individual phones in an utterance. In LID, grammars equivalent to a bigram grammar have generally been employed, i.e., these models capture the likelihood that each phoneme is followed by any other phoneme. The language of an utterance is determined by successively decoding it with the unit models and grammar of each of the target languages. The decoding with the highest likelihood is taken to indicate the language in which the utterance was spoken. Since the likelihood score computed during the decoding process is a product of both acoustic and grammar scores, it actually incorporates both acoustic and phonotactic information.

*LID with Language-Dependent Acoustic Models*

The most successful LID systems train a separate stochastic model for each phoneme in each of the target languages [113]. Let $L = \{L_1, L_2, ..., L_M\}$ be the set of languages to be identified. The approach based on language-dependent phone recognition uses a bank of $M$ parallel phone recognizers followed by phonotactic bigram models ($M$ models per phone recognizer) as shown in Figure 7. Such a system is also known as a parallel-phone-recognition-followed-by-language-model (PPRLM) system. A language-dependent score, $\ell_m$, is obtained as the sum of two terms:

$$\ell_m = \log P(x|L_m) + \sum_{j=1}^{M} \log P(\Phi_j|L_m) \tag{4.2.2}$$

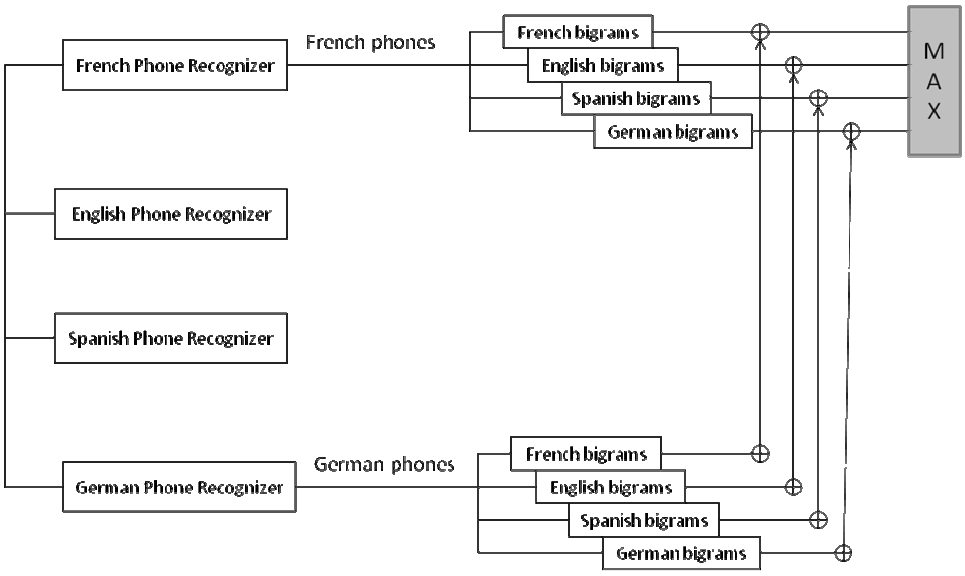



**Figure 7:** A block diagram of a language-dependent LID system for four languages.

where $x$ is the acoustic representation of the test utterance, $\Phi_j = \{\phi_1, \phi_2, ..., \phi_{T_j}\}$ is the phone sequence output from the $L_j$ phone recognizer, and $T_j$ is the number of recognized symbols. The first term in (4.2.2) is the output acoustic log-likelihood of the $L_m$ phone recognizer. The average log-likelihood over all utterances decoded by this phone recognizer is subtracted from this score to remove any bias between phone recognizers. The second term in (4.2.2) is the phonotactic score, computed from phone string output by the $M$ phone recognizers. The phonotactic score is the sum of the corresponding log-probabilities (normalized by the number of phones in the utterance) for each of the $M$ phone bigram models. The incoming test utterance is decoded by all language-dependent phone recognizers. The language with the highest $\ell_m$ score is hypothesized.

*LID with Language-Independent Acoustic Models*

An important simplification is made possible by noting that one can build a stochastic grammar for each language based on the acoustic models of one general acoustic model [60]. A universal set of acoustically defined, rather than linguistically defined, units is used to cover the acoustic characterization of the intended spoken

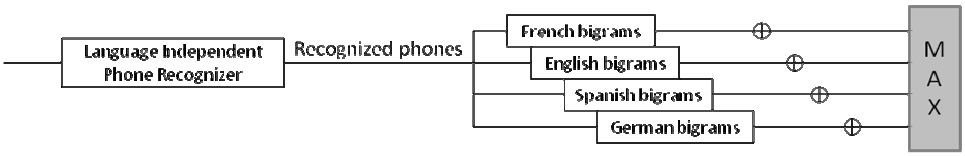



**Figure 8:** A block diagram of a language-independent LID system for four languages.

languages. The phone sequence output by a single language-independent phone recognizer is then rescored with language-dependent phonotactic models approximated by phone bigrams, as illustrated in Figure 8. For each language, a bigram model is estimated using the labels output by the language-independent phone recognizer for that language. The language providing the highest log-probability is hypothesized.

#### *4.2.2.3 Approaches to LID with Prosodic and Duration Information*

It has long been recognized that prosodic information (that is, information derived from speech characteristics such as pitch, amplitude, and rate, which span several phonemes) should contribute much to speech recognition. This insight has, however, not contributed much to the success of current systems. Similarly, the incorporation of explicit prosodic information was not as useful in early LID systems as the designers may have hoped [35, 75, 76], but recent research has begun to fulfill this hope [38]. Still, LID approaches based solely on prosodic information are rare. See [39] for an example where fundamental frequency and energy contours were used for LID.

In [35], pitch information was incorporated by multiplying the acoustic and phonotactic probabilities by a third factor that captured the probability densities of pitch distributions in the various languages. In [75], a more complex approach was taken, which took into account the pitch variation within and across the different segments marked by a broad-category (i.e., nasal, fricative, etc.) classifier. These prosodic features were found to be only marginally useful.

Segmental duration has been much more useful in characterizing language differences. In [35, 77], the distributions of the durations in each broad category were

modeled as additional factors in the computation of language likelihoods and this was observed to be quite useful in both cases.

#### *4.2.2.4 A Comparison of LID Approaches*

Formal evaluations have indicated that the most successful approach to LID relies on using the phonotactic content of a speech signal to discriminate among a set of languages. Although phone-based systems provide the best LID performance, their heavy computational demands may preclude their use in low cost, real-time applications. They also require phonetically labeled training data for each target language. Language-independent LID is the most practical implementation with respect to decoding and training, in particular to simplify extension to more languages.

GMM LID systems have significant potential advantages over phonotactic approaches. They do not require orthographically or phonetically transcribed speech and are far more computationally efficient. However, performance of GMM-based systems using acoustic scores has tended to be significantly worse than that of the phonotactic approaches [113].

Phonotactic LID is still an active research area with many problems and challenges. The first challenge is posed by the recognition (decoding) process of the sound units from continuous speech, which often needs to be performed in adverse acoustic conditions. The second challenge is to achieve modeling of a sufficiently high statistical order. Language-specific dependencies may be found that span several phonemes and the modeling accuracy grows with the model order. On the other hand, the model complexity in terms of the number of free parameters grow exponentially with its order.

### 4.2.3 LID Corpora

On the basis of the work by Muthusamy et al., the first multilingual database for LID research was the Oregon Graduate Institute Multi-Language Telephone Speech

(OGI-11L) corpus [78]. It was released in 1993 and has been used extensively for the evaluation of LID systems. The training partition of the corpus consists of monologue speech collected from 11 languages, with a total of about 90 minutes per language from about 100 different speakers per language. Each speaker contributed between one and two minutes of speech. The languages were English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil, and Vietnamese.

In a subsequent OGI project, multilingual data was collected from at least 200 speakers in 22 different languages [58]. The languages include Arabic, Cantonese, Czech, English, Farsi, French, German, Hindi, Hungarian, Italian, Japanese, Korean, Malaysian, Mandarin (Chinese), Polish, Portuguese, Russian, Spanish, Swahili, Swedish, Tamil, and Vietnamese. This database was released in 1995.

In 1996 and 1997, the Linguistic Data Consortium (LDC) [59] organized a series of LID-relevant data collections of unscripted telephone conversations between friends in six languages (CallHome) or in 12 languages (CallFriend). There are about 60 conversations per language each lasting five to 30 minutes.

The CallFriend corpus is a collection of unscripted conversations for 12 languages recorded over domestic telephone lines. The target languages are Arabic, English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil, and Vietnamese. Three of the 12 languages (English, Mandarin, and Spanish) contain material for two dialects. The CallFriend corpus was used by National Institute of Standards and Technology (NIST) as a source to conduct an evaluation of LID systems.

### 4.2.4 Feature Extraction for LID

The most common feature sets used for LID are mel-frequency cepstral coefficients (MFCC), perceptual linear prediction (PLP) features, and shifted delta cepstra (SDC) features.

MFCCs are the most common features in the LID community as they have become standard in many modern ASR architectures. They are an effective means to model the human perception of speech signals. For obtaining MFCCs, the frequency bands are positioned logarithmically in the mel frequency domain to approximate the human auditory system's response closely.

Perceptual linear prediction (PLP) modifies the short-term spectrum of the speech by several psychophysically based transformations as is the case with MFCC feature extraction. For obtaining MFCCs, the spectrum is warped according to the mel scale, whereas for obtaining PLP features the spectrum is warped according to the Bark scale. The Bark scale is linear at low frequencies and logarithmic at high frequencies. PLP reduces the sensitivity of ASR systems to changes in high frequencies and increases the sensitivity to the changes in the first and second formants.

A previous study [9] showed that improved LID performance could be obtained by using SDC feature vectors created by stacking delta cepstral features computed across multiple speech frames. The SDC features incorporate additional temporal information about the speech into the feature vectors [101].

## *4.3 MOP in Statistical Learning Problems*

MOP has been useful in many engineering problems ranging from chemical engineering to aerospace engineering. Adoption of MOP for machine learning and pattern recognition problems is fairly new, e.g. see [29, 34] for surveys on the use of MOP in bioinformatics and data mining fields, and this section is devoted to an overview of some exemplary work.

### 4.3.1 The Conventional Overall-Objective Approach

In the literature, by far the most used approach to solve a multi-objective problem consists of transforming it into a single-objective problem. Traditional algorithms aim to satisfy multiple objectives by forming a global objective function and solving the

resulting problem through the use of classical single-objective programming (SOP) methods. Many practical approaches to MOP problems adopt an overall objective function which is a weighted-sum of the arising conflicting objectives, e.g.,

$$f(\Lambda) = \sum_{k=1}^{K} \alpha_k f_k(\Theta) \tag{4.3.1}$$

where $\alpha_k, k = 1, ..., K$, denotes the weight assigned to criteria $f_k$. For instance, in 2005 NIST LRE the overall objective function was defined as in (2.2.22), which was roughly the average of the false-alarm and miss rates for the different target languages involved.

Combining several competing objectives into an overall objective function, such SOP-based approaches promise that the chosen overall objective function is optimized. However, there is no guarantee on the performance of the individual objectives as they are not considered separately. Moreover, one or more of these conflicting objectives tend to dominate the optimization process. It will easily become overwhelming for the designer to find an overall objective function that achieves desirable levels for the individual objectives. For these reasons, we articulate that methods of traditional SOP are not enough and take an MOP perspective for solving such problems.

### 4.3.2 Use of Meta-Heuristic MOP in Pattern Classification

Optimization methods could be divided into derivative and non-derivative methods as shown in Figure 9 [1]. The non-derivative methods do not require any derivatives of the objective function in order to calculate the optimum. They are quite well-researched and are reported to find a global optimum. On the other hand, they are black box methods which require some critical information from user so that they generate intermediate steps.

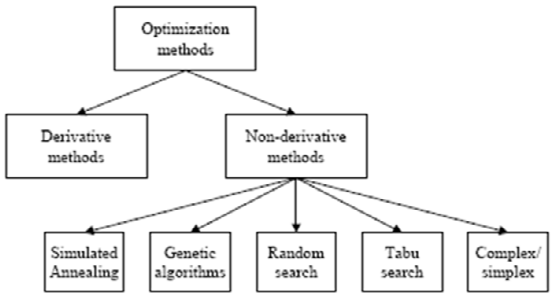



**Figure 9:** The non-derivative MOP methods.

#### *4.3.2.1 Genetic Algorithms (GAs)*

The basic idea of GAs [19,36] is the mechanics of natural selection. Each optimization parameter, $(w_k)$, is coded into a gene as, for example, a real number or string of bits. The corresponding genes for all parameters, $w_1, w_2, ..., w_M$, form a chromosome, which describes each individual. A chromosome could be an array of real numbers, a binary string, a list of components in a database, depending on the specific problem. Each individual represents a possible solution, and a set of individuals form a population. In a population, the fittest are selected for mating. Mating is performed by combining genes from different parents to produce a child, called a crossover. Finally the children are inserted into the population and the procedure starts over again, thus representing an artificial Darwinian environment. The optimization continues until the population has converged or the maximum number of generations has been reached.

As GAs have be around for such a long time they also have the broadest field of applications. GAs are however associated with a high computational cost. Moreover, GAs are more complicated and harder to implement and parameterize than other methods.

[99] considers the adaptation of kernel and regularization parameters of support vector machines (SVMs) [104] by means of multi-objective evolutionary optimization.

Support vector machines are reviewed from the multi-objective perspective, and different encodings and model selection criteria are described. The three objectives in the context of the detection of pedestrians in infrared images for driver assistance systems are the minimization of the false positive rate, the false negative rate, and the number of support vectors to reduce the computational complexity.

One of the most researched engineering problems from an MOP point of view is regularization. From the multi-objective optimization point of view, including a regularization term in the cost function is equivalent to combining two objectives using a weighted aggregation formulation. Thus, it is straightforward to re-formulate the regularization techniques as multi-objective optimization problems. [64] presents an algorithm for learning with neural networks [73] based on MOP. The authors consider three performance objectives, namely, the Euclidean distance and maximum difference measurements between the real nonlinear system and the nonlinear model to increase the robustness to learning. They select a subset from a large set of basis functions using GAs. This is equivalent to model selection.

[52] presents a method for regularizing neural networks using multi-objective evolutionary algorithms. No hyperparameter needs to be specified beforehand. A number of Pareto-optimal neural networks, instead of one single network were generated in the evolutionary optimization. They show that neural network regularization can be addressed from the multi-objective optimization point of view. This approach exhibits two advantages over traditional regularization techniques. First, a number of neural networks of a spectrum of model complexity instead of one single neural network can be obtained in one optimization run. Second, a new and more direct regularizer can be used. Similar approaches are described in [11, 64]. All these previous work indicate that MOP offers a great degree of freedom for obtaining a proper tradeoff among accuracy and model complexity.

#### *4.3.2.2 Simulated Annealing (SA)*

SA was first presented by Kirkpatrick [50] in the early 80s. SA simulates the natural phenomena of annealing of solids in order to optimize complex systems. Annealing of solids are accomplished by heating up a solid and allowing it to cold down slowly so that thermal equilibrium is maintained. This ensures that the atoms are obtaining a minimum energy state. The algorithm starts with an initial design. New designs are then randomly generated in the neighborhood of the current design according to some algorithm. The change of objective function value, ($\triangle E$), between the new and the current design is calculated as a measure of the energy change of the system. If the new design is superior to the current design ($\triangle E < 0$) it replaces it, and the procedure starts over again. An advantage of SA is that it can handle mixed discrete and continues problems. The parameters settings for a SA algorithm determines how new solutions should be generate, the initial temperature and what the cooling scheme should be.

#### *4.3.2.3 Tabu Search (TS)*

TS [32] is an adaptive heuristic strategy that was primarily designed for combinatorial optimization. In the TS method, flexible memory cycles (tabu lists) are used to control the search. At each iteration we take the best move possible that is not tabu, even if it means an increase in objective function value. The idea is that when we reach a local minimum we wish to escape via a different path.

There is no simple answer to which optimization methods is the best for any given problem. It is all a matter of opinion depending on the nature of the problem and the availability of different optimization software that fits the problem statement. In most comparison studies different methods come out on top depending on the problem and how well the different methods have been tuned to fit that particular problem.

SA and TS are growing in popularity and gaining ground on GAs mostly on

combinatorial optimization problems. SA could actually be seen as a subset of GA's with a population of one individual and a changing mutation rate. Both SA and TS are robust methods slightly less computational expensive then genetic algorithms.

#### *4.3.2.4 Random Search (RS)*

RS is a generic term for methods that rely on random numbers to explore the search space. Random search methods are generally easy to implement, and depending on the implementation they can handle mixed continues and discrete problems. However, they usually show quite slow convergence.

### 4.3.3 Evaluation Function Development

MOP may allow one to learn about the trade-offs in a problem and to identify recurring patterns in the Pareto fronts, the knowledge and understanding of which may help in the formulation of novel and better single-objective problem formulations. In the meantime, the preferences can directly be entered in the problem statement instead of trying to find a way to combine all of them into one single overall preference function.

### 4.3.4 Visualization and Solution Identification

The large majority of MOPs identified in this review have been tackled by generating a whole Pareto front and by then applying (or hoping to apply) some form of decision making process afterwards to choose a single solution. This strategy defers decision making until "all the information is in" (a good thing when little is known about the possible trade-offs) but the problem remains how to identify/select a single best solution. The really successful application of this mode of MOP thus calls for advanced methods for the visualization of the Pareto front and for the support of the decision-maker in selecting solutions from it. Evidently, straightforward visualizations of the Pareto front are only possible in two or three dimensions, and a representation of the

solutions obtained and the relationships between them becomes much more intricate for higher dimensions.

To date, only few methods for effective visualization have been introduced that can deal with the truly multidimensional case (one of the main examples is a parallel axis plot [96]), and visualization remains a major topic for future research. Automatic identification of promising solutions from Pareto front approximations has been investigated in several recent works [70]. However, these papers have generally dealt with methods for steering/focusing the search towards the (potentially) more important areas without the need for additional preference information from the decision-maker (usually by searching more strongly in regions of the Pareto front that have highest local curvature).

An alternative approach is to first obtain the most complete Pareto front approximation set possible and then to reduce this set to a single solution by some automated process, taking account of the whole Pareto front shape and other information. Where expert knowledge on how to balance conflicting measures/goals is available, this can be extracted by using preference articulation techniques [49], whereby a series of concrete questions about preferences are asked to the DM. The answers then determine if it is possible to build one or other type of consistent model of the DM's internal utility function. If so, then an automated procedure can potentially be developed for solution evaluation/selection. Note that far more complicated types of model exist than a simple weighted sum over the objectives. These preference articulation techniques are so far used mainly in operations management and have yet to be transferred to other applications, but there is plenty of potential for future successes in this area.

# CHAPTER V

# ITERATIVE CONSTRAINED OPTIMIZATION (ICO)

In this chapter, we develop a framework to iterative learn the parameters of a statistical technique, called iterative constrained optimization (ICO). The key motivation in ICO is that it is possible to reduce the value of an objective whenever the value of a conflicting objective is increased. Consider simultaneously minimizing the false-acceptance and false-rejection rates (or equivalently, the false-alarm and miss rates) in a detection problem. The Neyman-Pearson criterion [79] says that we should construct our decision rule to have minimum probability of miss while not allowing the probability of false alarm to exceed a certain value $\alpha$. In other words, the optimal detector according to the Neyman-Pearson criterion is the solution to the following constrained optimization problem:

$$\min P(\text{Miss}) \text{ subject to } P(\text{False-alarm}) \leq \alpha \tag{5.0.2}$$

Similarly, in ICO, each one of the objectives is iteratively optimized one after another with constraints on others, and the constraint bounds are adjusted by using the objective functions attained in the most recent iterate. It then becomes possible to tradeoff the performance in already-good objectives to improve the remaining not-so-good objectives. By doing so, a better balance among many competing objectives can be achieved where each objective is comparably good.

Let $\Theta = \{\mathbf{w}_1, ..., \mathbf{w}_M\}$ denote a set of decision vectors that we want to optimally choose in a learning task. Suppose that we are given a set of $K$ competing objectives, $f_k(\Theta) \in (0, 1),\ k = 1, ..., K$, each of which is a nonlinear function of the decision vectors. The *best compromise solutions*, $\hat{\Theta} = \{\hat{\mathbf{w}}_1, ..., \hat{\mathbf{w}}_M\}$, are found by MOP, which

is formulated as

$$\hat{\Theta} = \{\hat{\mathbf{w}}_1, ..., \hat{\mathbf{w}}_M\} = \arg\min_{\Theta} \left[f_1(\Theta), f_2(\Theta), ..., f_K(\Theta)\right]. \tag{5.0.3}$$

## 5.1 ICO

Consider formulating the multi-objective programming (MOP) problem in (5.0.3) as a set of $K$ single-objective programming (SOP) problems in the form

$$\begin{aligned} \min_{\theta} \quad & f_k(\theta) \\ \text{subject to} \quad & f_p(\theta) \leq \overline{f}_p,\ p = 1, ..., K, p \neq k, \end{aligned} \tag{5.1.1}$$

for all $k = 1, ..., K$, i.e., the minimization of one objective function, $f_k(\theta)$, with proper constraints on the other $(K-1)$ competing objectives, $f_p(\theta)$, where $\overline{f}_p$'s are the upper bounds for the objective functions to be attained. The tightness of the constraints specify the size of the search region. On one hand, loose constraints may result in more degradation in the other objectives than desired. On the other hand, tight constraints do not change the feasible region much and hence might not yield the desired change in the individual objectives.

### 5.1.1 Generating the Constraints

Starting with an iterate, $\Theta^{(1)}$, and an objective vector, $f^{(1)} = (f_1^{(1)}(\Theta), ..., f_K^{(1)}(\Theta))$, the goal in ICO is to move into another iterate, $\Theta^{(2)}$, yielding an objective function vector $f^{(2)}$ where at least one objective function $f_k, 1 \leq k \leq K$ attains a *considerably improved* value, while others are *possibly slightly degraded*.

Consider the illustration in Figure 10 for a two-objective optimization problem, where point $Z_0$ is the starting point, and a better compromise solution is being searched for. Due to the conflicting nature of two objectives, it is possible to achieve a reduction in $f_1$ when $f_2$ is allowed to slightly increase. To compensate for the performance loss in $f_2$, $f_1$ is allowed to slightly increase, and the best $f_2$ for the given $f_1$ value is searched for.

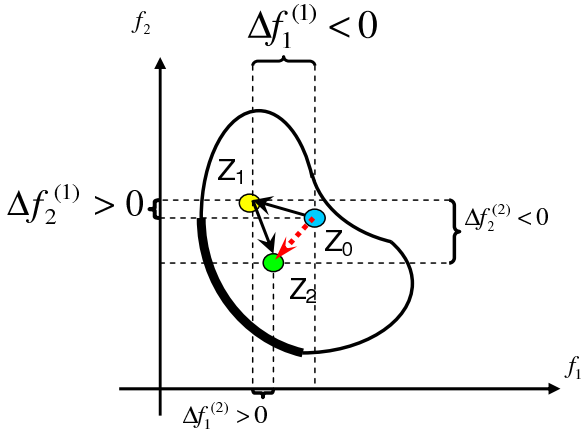



**Figure 10:** An illustration of the search for a better compromise solution in a close neighborhood of the current compromise solution.

Note that the vector of objective functions in one iteration is slightly perturbed, and set as the constraint bounds in the next iteration. This corresponds to

$$\bar{\mathbf{f}} = \mathbf{f} + \delta. \tag{5.1.2}$$

where $\mathbf{f}$ is the vector of the most recent objective function values, and $\delta \in \Re^K$ is a vector of small perturbations added to $f$.

### 5.1.2 Finding and Validating the Step-Size

It is important to note that, in general, it is not necessarily correct that the resulting individual objectives are preferable to the initial ones, especially when there are many competing objectives: What is gained in one iteration can quickly be lost in the subsequent iterations. With the increasing number of objectives, it becomes essential to control changes from one iteration to the next. This is because when one objective is increased, some of the other objectives reduce whereas some may increase. The step-size control strategy of ICO is application-specific. See Section 6.2.4.1 for LM training and adaptation and Section 7.2.2.2 for the LID application.

The step-size control strategy is motivated by the fundamental concepts known as *preference* and *indifference* of utility theory [26]. Although it is beyond the scope of this thesis, the principle behind utility theory is quite simple: It is to assign to each

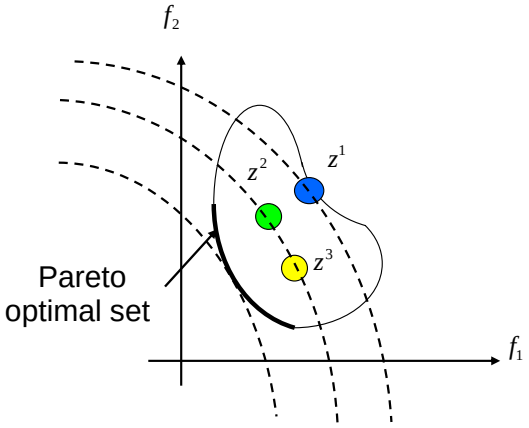



**Figure 11:** Indifference curves for a user's preference model.

element of a set a real number in such a way that the higher the number, the more preferred the element. In utility theory, an individual's preferences are represented by a utility function, $u(.)$. Consider three elements $z^1$, $z^2$ and $z^3$ in a set. When the individual prefers element $z^2$ to element $z^1$, it is said that $z^2$ *is preferred to* $z^1$, which is denoted with the $\succ$ symbol as in (5.1.3). In this case, the utility of $z^2$ is greater than that of $z^1$. When the two elements $z^2$ and $z^3$ are equally preferable or an individual does not have any preference for one of them, then it is said that $z^2$ *and* $z^3$ *are indifferent*. This latter relation is denoted with a $\sim$ symbol as in (5.1.4). In this case, the utilities of $z^2$ and $z^3$ are the same. A model of the individual's preference logic can then be constructed using $u(.)$ in the following way:

$$z^2 \succ z^1 \text{ iff } u(z^2) > u(z^1), \tag{5.1.3}$$

$$z^2 \sim z^3 \text{ iff } u(z^2) = u(z^3). \tag{5.1.4}$$

When the negative of a cost function is chosen as a utility function, the point where the cost function attains a lower value is of better utility. This situation is illustrated in Figure 11, where $z^2$ and $z^3$ are of the same utility and equally preferable, and both are of better utility than $z^1$.

### 5.1.3 Fair Comparison with SOP-based Techniques

In this section, we address the issue of comparing the MOP-trained classifiers with the more traditional SOP-trained classifiers. Since all of the conflicting objectives are important to a system designer, no single overall performance measure should be taken as a basis for a realistic comparison. Nevertheless, any new method is expected to provide some advantage over existing approaches. For instance, the new method may reduce the computational complexity or the storage memory requirements. In many applications, a new method is expected to improve some performance measure such as average classification error rate in the LID application and WER in LM training and adaptation.

For many realistic classification tasks, we typically want to prevent bias towards any of the objectives and desire a symmetry across the many competing objective functions. If the design goal is to achieve a balance among several objectives, a realistic evaluation of classifiers can be achieved by comparing the range of values that the individual objective functions take on. ICO-trained classifiers offer flexibility to design a system that yields a narrower range for the individual objectives than the SOP-trained classifiers. A narrow range of objective function values mean that each of the objective function has taken reasonable consideration in the optimization process so that all objectives attain a somewhat comparable values. On the contrary, a wide range of values mean some objective functions have stolen from the share of some other objectives. As a corollary, the ICO framework is promising for producing less *outliers* in the objective function space compared to the SOP algorithms with an overall objective function. This is especially important when there are more competing objectives than just two. One way to quantify the degree to which a classifier results in outlier objective values is to compare the upper and lower 5% or 10% percentile averages.

## *5.2 Theoretical Properties of ICO*

In this section, we investigate theoretical properties of the ICO method including the Pareto optimality of the solutions, the tradeoff rates among the objectives at a solution, the factors needed to be considered for generating the constraint bounds, and the decision-making for stopping the search.

### 5.2.1 Pareto Optimality of the Solutions Generated by ICO

**Theorem 5.2.1** *The solution to the ICO method is Pareto optimal for any given upper bound for the constraints.*

The Pareto optimality of an ICO solution is examined by analyzing the optimality of the solution to the problem in (7.2.3) using the Karush-Kuhn-Tucker (KKT) conditions for the Lagrangian function

$$L_k(\Theta, \lambda_k) = f_k(\Theta) + \sum_{i \neq k} \lambda_{ki}(f_i(\Theta) - \bar{f}_i). \tag{5.2.1}$$

**Theorem 5.2.2** *The KKT optimality conditions for this problem can be written as [7, 80]*

$$(1) \quad \nabla L(\Theta^*, \lambda^*) = 0, \tag{5.2.2}$$

$$(2) \quad f_i(\Theta^*) \geq \bar{f}_i,\ i \in \Im, \tag{5.2.3}$$

$$(3) \quad \lambda_i^*(f_i(\Theta^*) - \bar{f}_i) = 0,\ i \in \Im, \tag{5.2.4}$$

$$(4) \quad \lambda_i^* \geq 0,\ i \in \Im \tag{5.2.5}$$

The KKT conditions require that we have

$$(1)\nabla f_k(\Theta^*) + \sum_{i \neq k} \lambda_{ki}^* \nabla f_i(\Theta^*) = 0 \tag{5.2.6}$$

$$(2)\lambda_{ki}^*(f_i(\Theta^*) - \bar{f}_i) = 0, \tag{5.2.7}$$

$$(3)\lambda_{ki}^* \geq 0, i = 1, ..., K, i \neq k. \tag{5.2.8}$$

The complementary slackness condition in (5.2.7) requires that one of the following two conditions holds for the constraints:

1. None, one or more of the constraints in (7.2.3) are non-binding constraints for which we have

$$\lambda_{ki}^* = 0, \text{ if} f_i(\Theta^*) \neq \bar{f}_i, \tag{5.2.9}$$

   i.e., the $i^{th}$ objective needs not be at its maximum allowed level for minimizing $f_k$ as in (7.2.3).

2. One or more of the constraints in (7.2.3) are binding constraints for which we have

$$\lambda_{ki}^* > 0, \text{ if} f_i(\Theta^*) = \bar{f}_i, \tag{5.2.10}$$

   i.e., the $i^{th}$ objective needs to be at its maximum allowed level so that the $k^{th}$ objective can be minimized. At the optimal solution these binding constraints are at the constraint bound level, i.e.,

$$f_i(\Theta^*) = \bar{f}_i. \tag{5.2.11}$$

Note that there has to be at least one objective for which the corresponding constraint in (7.2.3) is binding. This comes from the conflicting nature of the objectives: If none of the objectives were binding, we could reduce the primary objective even further until until at least one of the constraints becomes binding.

The optimal value of the Lagrangian function can be rewritten by plugging (5.2.7) into (5.2.1) as

$$L(\Theta^*, \lambda^*) = f_k(\Theta^*), \tag{5.2.12}$$

i.e., $L(\Theta^*, \lambda^*)$ is equal to the value achieved by the primary objective function, $f_k(\Theta^*)$. Note that the Lagrange multipliers are equal to the negative of the partial derivatives of the Lagrange function, $L_k$, with respect to the constraint bounds, $\bar{f}_i$, i.e.,

$$\lambda_{ki} = -\frac{\partial L_k(\Theta, \lambda)}{\partial \bar{f}_i}. \tag{5.2.13}$$

Using (5.2.12), the optimal Lagrange multipliers, $\lambda^*_{ki}$, can be rewritten as

$$\lambda^*_{ki} = -\frac{\partial f_k(\Theta^*)}{\partial \bar{f}_i}. \tag{5.2.14}$$

Furthermore, we can plug (5.2.11) into (5.2.14) and obtain

$$\lambda^*_{ki} = -\frac{\partial f_k(\Theta^*)}{\partial f_i(\Theta^*)}, \tag{5.2.15}$$

i.e., the Lagrange multipliers, $\lambda^*_{ki}$, quantify the trade-off rates between $f_k$ and $f_i$ at the optimal point, $\Theta^*$. The optimal Lagrange multipliers indicate how much we need to increase $f_i$ to reduce $f_k$ or equivalently, how much we would gain in $f_k$ by increasing $f_i$. Thus, we can make the following two conclusions:

1. If $\lambda^*_{ki} = 0$, the corresponding constraint is non-binding at the optimal solution. It means that we do not need to increase $f_i$ anymore to reduce $f_k$. Hence, there is not a tradeoff concerning $f_i$ and $f_k$ at the optimal point, $\Theta^*$.

2. If $\lambda^*_{ki} > 0$, the corresponding constraint is binding at the optimal solution. It means that any further increase in $f_i$ will reduce $f_k$ attained at the optimal point, $\Theta^*$. Hence, there is a tradeoff concerning $f_i$ and $f_k$ at $\Theta^*$.

The definition of Pareto optimality dictates that a solution is Pareto optimal if we need to increase at least one objective, $f_i$, to reduce any of the other objectives, $f_k$. The above conclusions reveal that we need to increase those objective functions for which $\lambda^*_{ki} > 0$ to reduce the primary objective function, $f_k$. Since there is always at least one such objective for which $\lambda^*_{ki} > 0$, at least one objective, $f_i, i \neq k$ should be increased to reduce $f_k(\Theta^*)$. Therefore, the solution $\Theta^*$ is a Pareto optimal solution with the tradeoff rates, $\lambda_{ki}$ as found in (5.2.15).

### 5.2.2 Test of a Solution for Pareto Optimality

We can also test the Pareto optimality of any given solution. A popular way of checking Pareto optimality is given by the next theorem, adopted from [].

**Theorem 5.2.3** *Let $\hat{\Theta}$ be a given solution. Solve the problem:*

$$\begin{aligned} max \quad & z = \sum_{i=1}^{k} \epsilon_i \\ subject\ to \quad & f_i(\Theta) + \epsilon_i \leq f_i(\hat{\Theta}), i = 1, ..., k \\ & \epsilon_i \geq 0, i = 1, ..., k, \end{aligned}$$

*where both $\Theta$ and $\epsilon$ are variables. Then, the following results are valid.*

- *The vector $\hat{\Theta}$ is Pareto optimal if and only if this problem has an optimal objective function value of 0.*
- *If this problem has a finite non-zero optimal objective function value at $\bar{\Theta}$, then $\bar{\Theta}$ is Pareto optimal.*

Note that $\epsilon_i$ in the above theorem quantifies how much each objective function, $f_i(\Theta)$, deviates from $f_i(\hat{\Theta})$ Moreover, the value of the objective function, $z$, is the sum of these differences. Thus, the maximum of $z$ is the sum of the maximum deviations of the objective functions from the values $f_i(\hat{\Theta})$. The first item in this theorem states that if the maximum of $z$ is 0, the point $\hat{\Theta}$ is a point where any of the objectives cannot be reduced without increasing others, i.e., $\hat{\Theta}$ is a Pareto optimal solution. Otherwise, $\hat{\Theta}$ is still Pareto optimal if the maximum of $z$ is non-zero but the sum of the gradients of the objectives vanishes.

Based on this theorem, we should solve the following problem:

$$\begin{aligned} max \quad & \sum_{k=1}^{K} \epsilon_m = z \\ s.t. \quad & f_k(\Theta) + \epsilon_k \leq f_m(\hat{\Theta}), k = 1, ..., K \\ & \epsilon_k \geq 0, k = 1, ..., K \end{aligned}$$

The Lagrange function for this problem is:

$$L(\epsilon, \Theta, \lambda) = \sum_{k=1}^{K} \epsilon_k - \sum_{k=1}^{K} \lambda_k(\epsilon_k + f(\Theta) - f(\hat{\Theta}))$$

The optimal solution $\epsilon^*, \Theta^*, \lambda^*$ is found by equating the derivatives of the Lagrangian function with respect to those parameters to 0. We then obtain:

$$(1) \qquad \frac{\partial L}{\partial \epsilon_k} = 1 - \lambda_k^* = 0 \Rightarrow \lambda_k^* = 1. \tag{5.2.16}$$

$$(2) \qquad \frac{\partial L}{\partial \Theta_k} = -(\epsilon_k + f_k(\Theta^*) - f_k(\hat{\Theta})) = 0 \Rightarrow \epsilon_k^* = f_k(\hat{\Theta}) - f_k(\Theta^*) \tag{5.2.17}$$

$$(3) \qquad \nabla_{\mathbf{w}_k} L = -\sum_{m=1}^{M} \nabla_{\mathbf{w}_k} f_m = 0, \tag{5.2.18}$$

for all $k = 1, ...., M$. From (5.2.16) and (5.2.17), we obtain the optimal objective function value as

$$z^* = \sum_{m=1}^{M} \epsilon_m^* = \sum_{m=1}^{M} E_m(\hat{\Theta}) - \sum_{m=1}^{M} f(\Theta^*).$$

That is, the optimal objective function value is the sum of the objective values at the given solution $\hat{\Theta}$ less the sum of objectives at the optimal solution $\Theta^*$.

### 5.2.3 Generating the Constraint Bounds

We can infer the tradeoff rates at any solution, $\Theta^*$, using (5.2.15). The optimal Lagrange multipliers, $\lambda_{ki}^*$ indicate how much we should give up from $f_i(\Theta^*)$ to minimize $f_k(\Theta^*)$. If a Lagrange multiplier, $\lambda_{ki}^*$, is large, it means that a small increase in $f_i$ will result in a significant reduction in the primary objective, $f_k$. In contrast, a small Lagrange multiplier, $\lambda_{ki}^*$, indicates that only a small reduction in the primary objective, $f_k$, can be obtained by increasing $f_i$. Note that a decision maker should decide how much one should compromise any objective to reduce others. Therefore, the decision maker implicitly or explicitly specifies what is a small and what is a large value for the tradeoff rates.

The constraints for the next subproblem can be generated based on the optimal Lagrange multipliers of the most recent solution based on how much tolerance we have for each objective. In the applications that we consider in this thesis, the constraints

are generated by perturbing the most recent solutions by a small fixed fraction. Furthermore, the objectives are lexicographically ordered so that the objective for which the performance on a separate development set is worst comes first and so on. Instead of perturbing most recent solutions by a fixed amount and setting as constraint bounds, one can adjust the constraint bounds based on the optimal Lagrange multipliers. This strategy would offer the added advantage that the tradeoff rates would be effectively used.

### 5.2.4 Stopping at a Satisfactory Solution

A multi-objective problem is considered solved when one of these conditions hold:

- All of the Pareto optimal solutions have been found.
- A number of Pareto optimal solutions have been found so that the Pareto optimal curve can be estimated.
- At least one satisfactory Pareto optimal solution has been found.

In most problems, finding all of the Pareto optimal solutions is not possible either because they can be infinitely many or because of computational limitations. Therefore, it is often desirable to find a number of Pareto optimal solutions that are enough to estimate the Pareto optimal curve. After estimating the Pareto optimal curve, one can then select one of the found solutions as the final solution of the MOP solution or continue generating Pareto optimal solutions until a desirable one is found. In any case, whenever a solution is to be decided as the final solution, a decision maker has to be involved in the problem to define "a desirable final solution". In the applications that will be considered in the subsequent chapters, a solution is satisfactory if:

1. All the objectives (the false-acceptance and false-rejection rates in the detection of languages) attain comparable values in the LID application application.

2. Too much dependence on either of two models (where each model is derived using a separate dataset) is avoided in the LM adaptation application.

We can make the decision as to when to stop by evaluating the optimal Lagrange multipliers, $\lambda_{ki}^*$. If the Lagrange multipliers are all small, it is indicated that only small gain will be obtained even when all the objectives are significantly increased from their most recent values. Thus, such a condition would suggest stopping. In the meantime, even when the Lagrange multiplier is not small, we might not want to increase any one of $f_i$ to reduce $f_k$ and we may want to stop at such a point. It is also possible to reach a point where we cannot make much progress since the problem gets tighter at each iteration because of lower values attained for the individual objectives. Such points of insignificant progress are also points where one would be interested in stopping.

# CHAPTER VI

# STATISTICAL LANGUAGE MODELING AND ADAPTATION

In this chapter, we first develop an SOP-based technique for language modeling and adaptation, called structural maximum a posteriori (SMAP) probability estimation. We later develop an MOP-based alternative to SMAP. We motivate the use of the these techniques for LM adaptation scenario although they are valid for more general problem of language modeling. Finally, we report our experimental results to compare our SOP- and MOP-based methods.

## *6.1 An SOP-Based Baseline: SMAP*

In this section, we propose a structural maximum *a posteriori* (SMAP) framework for LM training and adaptation. There are three key steps in the formulation of the proposed SMAP approach. In the first step, $n$-gram trees are built to characterize the syntactic structure of the general domain data. There are as many trees as the unigrams in the text. In the second step, the hyperparameters of the prior probability distribution are hierarchically accumulated from the parent nodes. In the third step, the application-specific information is merged with the prior information to evaluate the SMAP $n$-gram probability estimates.

SMAP is based on the optimization of an overall training objective, and it is the reference system for the MOP-based system that we describe in Section 6.2. We later present key experimental results regarding the comparison of SMAP to the MOP-based approach in Section 6.3.

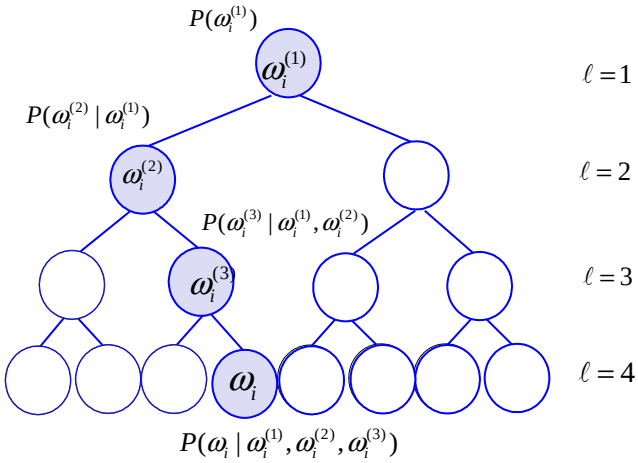



**Figure 12:** An illustration of embedding $n$-grams into the branches of tree structures.

### 6.1.1 Formulation of LM adaptation as an SOP Problem

Consider Figure 12 for estimating the 4-gram probability of the word $\omega_i$ at layer $\ell = 4$ given its history, i.e., words $\omega_i^{(1)}$ through $\omega_i^{(3)}$. The word at the root node, $\omega_i^{(1)}$, is said to be the *history node* of $\omega_i$ at the first layer and is denoted as $h_{\omega_i}^{(1)} = \omega_i^{(1)}$. The words $\omega_i^{(2)}$ and $\omega_i^{(3)}$ together with their own history nodes are the history nodes of $\omega_i$ at the second and at the third layer, respectively, i.e., $h_{\omega_i}^{(2)} = \{\omega_i^{(1)}, \omega_i^{(2)}\}$ and $h_{\omega_i}^{(3)} = \{\omega_i^{(1)}, \omega_i^{(2)}, \omega_i^{(3)}\}$. Thus, each tree branch corresponds to an $n$-gram and is represented by the history nodes $h_{\omega_i}^{(\ell-1)}$ in the first $\ell - 1$ layers followed by $\omega_i$ at the $\ell^{th}$ layer.

Let $\theta_{h_{\omega_i}}^{(\ell)}$ denote the $n$-gram probability of $\omega_i$ at the $\ell^{th}$ layer given its history $h_{\omega_i}^{(\ell-1)} = \{\omega_{i-n+1}, ..., \omega_{i-1}\}$, i.e.,

$$\theta_{h_{\omega_i}}^{(\ell)} \triangleq P(\omega_i | h_{\omega_i}^{(\ell-1)} = \{\omega_i^{(1)}, ..., \omega_i^{(\ell-1)}\}).$$

When an appropriate prior distribution, which is denoted as $g(\theta_{h_{\omega_i}}^{(\ell)})$, is known, $\theta_{h_{\omega_i}}^{(\ell)}$ can be calculated by maximizing the a posteriori probability given the observations, $W$, as follows:

$$\hat{\theta} = \arg\max_{\theta} \; P(\theta|W) \approx \arg\max_{\theta} \; P(W|\theta) g^{\rho}(\theta). \tag{6.1.1}$$

A non-negative number, $\rho$, is included in (6.1.1) to control the contribution of the prior density in the estimation process. This is useful since, for instance, the prior density $g(\theta)$ might not be reliable and we may be interested in discounting its importance. It is implied in (6.1.1) that the greater $\rho$ is, the more we are depending on the prior information. Specifically, when $\rho = 0$, the prior information is discarded, and when $\rho \to \infty$, only the prior information is used.

The problem in (6.1.1) is equivalent to solving the following problem:

$$\hat{\theta} = \arg\max_{\theta} \ [\log P(W|\theta) + \rho \log g(\theta)], \tag{6.1.2}$$

where the first term is the log-likelihood of the data for a given model, i.e.,

$$\log P(W|\theta) = \sum_{\omega_i \in W} c(h_{\omega_i}^{(\ell-1)}, \omega_i) \log \theta_{h_{\omega_i}}^{(\ell)}. \tag{6.1.3}$$

It is noted that the likelihood function here is expanded for the case of $\ell^{th}$ layer. The second term in (6.1.2) is the weighted log-prior probability of the LM parameters, which is the subject of the next section.

### 6.1.2 Hierarchical Priors

To evaluate the SMAP estimates of the $n$-gram probabilities, it is crucial to have an appropriate prior model, $g(\theta)$, for the $n$-gram probabilities. Our approach here is based on the natural hierarchical ordering of $n$-grams in tree structures. Doing so makes it possible to use information from lower order $n$-grams in calculating the $n$-gram probabilities.

#### *6.1.2.1 Hierarchical Priors for HMM Adaptation*

The hierarchical propagation of the prior information that we adopt in SMAP LM training and adaptation is inspired by previous works for speaker adaptation presented in [31, 37, 97, 98]. Before going into the details of SMAP for language modeling, we briefly review these works.

In [31], it was assumed that the joint probability distribution of the mixture weights, $\nu_1, ...., \nu_K$, for each mixture density in an HMM is a multinomial distribution. The prior information about the mixture weights, denoted as $g(\nu_1, ..., \nu_K)$ was modeled with a Dirichlet density (the conjugate prior density of the multinomial density [45]) with the hyperparameters $\phi_k > 0$ as

$$g(\nu_1, ..., \nu_K; \phi_1, ..., \phi_K) = \frac{1}{Z(\nu_1, ..., \nu_K)} \prod_{k=1}^{K} \nu_k^{\phi_k - 1} \tag{6.1.4}$$

where $\nu_1 + \nu_2 + ... + \nu_{K-1} + \nu_K = 1$ and $Z(.)$ is a normalization factor. Using the prior probability density in (6.1.4), a MAP framework was established for the adaptation of mixture weights in acoustic models. Recently, this framework has also been used for the adaptation of latent semantic analysis for word clustering [16].

To enhance the estimation method in [31], an SMAP method was proposed for the adaptation of HMM parameters [97]. This SMAP model incorporated a structure into parameter estimation so that the probability density functions for model parameters at one level were used as priors for those of the parameters at adjacent levels. Such priors estimated in a hierarchical manner are often referred to as *hierarchical priors* [97].

#### *6.1.2.2 Hierarchical Priors for LM Training and Adaptation*

We have adopted the prior density estimation mechanism discussed above. This was motivated by making an analogy between the distribution of probabilities of discrete words and the distribution of mixture weights within a mixture density. MAP-based LM training and adaptation problem is very similar to MAP-based estimation of the mixture weights of a Gaussian mixture model. For the case of $n$-grams, the underlying distribution is the multinomial density, for which the conjugate prior density is the Dirichlet density. Referring to (6.1.4), the prior distribution of the $n$-gram probabilities are

$$g(P(\omega_i | h_{\omega_i}^{(\ell-1)})) = g(\theta_{\omega_i}^{(\ell)} | \phi_{h_{\omega_i}}^{(\ell-1)}) \propto {\theta_{h_{\omega_i}}^{(\ell)}}^{[\phi_{h_{\omega_i}}^{(\ell-1)} - 1]}, \tag{6.1.5}$$

where $\phi_{h_{\omega_i}}^{(\ell-1)}$ is the hyperparameter of the prior distribution and the normalization term in (6.1.1) is absorbed into the $\propto$ notation. By plugging the log likelihood of the multinomial distribution (6.1.3) and the prior Dirichlet density (6.1.5) into (6.1.3), the logarithm of the posterior distribution, $\log P(\theta|W)$, is proportional to

$$\begin{aligned}
&\propto \sum_{\omega_i \in W} \sum_{h_{\omega_i}^{(\ell-1)}} \Big[ c(h_{\omega_i}^{(\ell-1)}, \omega_i) + \rho(\phi_{h_{\omega_i}}^{(\ell-1)} - 1) \Big] \log \theta_{h_{\omega_i}}^{(\ell)} \\
&= \sum_{\omega_i \in W} \sum_{h_{\omega_i}^{(\ell-1)}} (\phi_{h_{\omega_i}}^{(\ell)} - 1) \log \theta_{h_{\omega_i}}^{(\ell)} \\
&= \log \prod_{\omega_i \in W} \prod_{h_{\omega_i}^{(\ell-1)}} {\theta_{h_{\omega_i}}^{(\ell)}}^{\phi_{h_{\omega_i}}^{(\ell)} - 1}. \qquad (6.1.6)
\end{aligned}$$

Using (6.1.6), we obtain a recursive formula for estimating the hyperparameters associated with the node at the $\ell^{th}$ layer using the hyperparameters at the $(\ell-1)^{st}$ layer in a *top-down* manner as

$$\phi_{h_{\omega_i}}^{(\ell)} = c(h_{\omega_i}^{(\ell)}, \omega_i) + \rho(\phi_{h_{\omega_i}}^{(\ell-1)} - 1) + 1. \qquad (6.1.7)$$

With this Dirichlet distribution, the hyperparameters are propagated from the root nodes to the leaf nodes as $\phi_{h_{\omega_i}}^{(1)} \rightarrow \phi_{h_{\omega_i}}^{(2)} \rightarrow ... \rightarrow \phi_{h_{\omega_i}}^{(n)}$. Note the effect of the parameter $\rho$, which we refer to as the *forgetting factor*, in propagating the hyperparameters from the root nodes to leaf nodes. An event associated with a child node is boosted with how often its parent node is observed. This propagation mechanism is especially useful in combatting the data sparsity problem when using high order $n$-gram models because the child nodes can inherit reliable information from their parent nodes.

Following the motivation and derivation of empirical Bayes approach in [37], the hyperparameters for the root nodes, i.e., for $\ell = 1$, are estimated as

$$\phi_{h_{\omega_i}}^{(1)} = 1 + \epsilon c(h_{\omega_i}^{(1)}), \qquad (6.1.8)$$

where $0 < \epsilon \leq 1$ is a weighting coefficient. Note that when $\epsilon$ is small, the unigram observation frequencies are discounted.

### 6.1.3 SMAP LM Training

The Dirichlet prior density, coupled with the fact that the sum of the probabilities $\theta^{(\ell)}_{h_{\omega_i}}$ over all words $\omega_i$ in the dictionary, which is denoted as $\Omega_\omega$, is equal to 1, yields the following problem to be solved:

$$\begin{aligned} \max_{\theta} \quad & \sum_{\omega_i \in W} \sum_{h^{(\ell-1)}_{\omega_i}} \Big[ c(h^{(\ell-1)}_{\omega_i}, \omega_i) + \rho(\phi^{(\ell)}_{h_{\omega_i}} - 1) \Big] \log \theta^{(\ell)}_{h_{\omega_i}} \\ \text{s.t.} \quad & \sum_{\omega_i \in \Omega_w} \theta^{(\ell)}_{h_{\omega_i}} = 1. \end{aligned} \tag{6.1.9}$$

By incorporating a Lagrangian factor $\lambda$ for the constraint, (6.1.9) has a closed-form solution given by

$$\theta^{(\ell)}_{h_{\omega_i},SMAP} = \frac{c(h^{(\ell-1)}_{\omega_i}, \omega_i) + \rho(\phi^{(\ell)}_{h_{\omega_i}} - 1)}{c(h^{(\ell-1)}_{\omega_i}) + \sum_{\omega \in \Omega_{\omega_i}} [\rho(\phi^{(\ell)}_{h_{\omega_i}} - 1)]}. \tag{6.1.10}$$

The SMAP estimates of the $n$-grams are calculated in a top-down manner as $\theta^{(1)}_{h_{\omega_i},MAP} \rightarrow \theta^{(2)}_{h_{\omega_i},MAP} \rightarrow ... \rightarrow \theta^{(n)}_{h_{\omega_i},MAP}$. Note that when $\rho = 0$, no prior information is used and the estimates $\theta^{(\ell)}_{h_{\omega_i}}$ become the ML estimates:

$$\theta^{(\ell)}_{h_{\omega_i},ML} = \frac{c(h^{(\ell-1)}_{\omega_i}, \omega_i)}{c(h^{(\ell-1)}_{\omega_i})}. \tag{6.1.11}$$

### 6.1.4 SMAP LM Adaptation

SMAP LM adaptation is similar to the problem of SMAP LM training described above. The distinction in LM adaptation is simply that the application-specific data is used to compute the likelihood of data while the general domain data is used to compute the appropriate prior information.

Let the *tilde* symbol denote the quantities that are estimated using the application-specific data. The problem to be solved is

$$\begin{aligned} \max_{\tilde{\theta}} \quad & \sum_{\omega_i \in W} \sum_{h^{(\ell-1)}_{\omega_i}} \Big[ \tilde{c}(h^{(\ell-1)}_{\omega_i}, \omega_i) + \rho(\phi^{(\ell)}_{h_{\omega_i}} - 1) \Big] \log \tilde{\theta}^{(\ell)}_{h_{\omega_i}} \\ \text{s.t.} \quad & \sum_{\omega_i \in \Omega_w} \tilde{\theta}^{(\ell)}_{h_{\omega_i}} = 1, \end{aligned} \tag{6.1.12}$$

where $\tilde{c}(\cdot)$ denotes the count of the event inside the parentheses in the adaptation data set. By following the same procedure to solve the problem in (6.1.9), the SMAP estimates of the adapted $n$-gram probabilities are found as

$$\tilde{\theta}^{(\ell)}_{h_{\omega_i},SMAP} = \frac{\tilde{c}(h^{(\ell-1)}_{\omega_i},\omega_i) + \rho(\phi^{(\ell)}_{h_{\omega_i}} - 1)}{\tilde{c}(h^{(\ell-1)}_{\omega_i}) + \sum_{\omega\in\Omega_\omega}[\rho(\phi^{(\ell)}_{h_{\omega_i}} - 1)]}. \tag{6.1.13}$$

## *6.2 An MOP-Based Approach: ICO for LM Adaptation*

The overall objective function of SMAP proposed in Section 6.1 is a composition of multiple objective functions: SMAP is concerned with the maximum likelihood estimation of the adaptation data as well as an appropriate representation of the prior information obtained from a general domain corpus. We can indeed separate the overall objective function of SMAP into its component objective functions (which are at least partially conflicting).

In this section, we propose an MOP-based approach to the language modeling and adaptation task. The proposed approach takes its roots from the ICO framework proposed in Chapter V. The problem is solved in an iterative manner such that each objective is optimized one after another with constraints on the others. Solved this way, the target LM turns out to be in the form of a log-linear interpolation of component LMs as in Section 4.1.7.2. The weights in this interpolation are found so that the similarity of the target LM to each of the component LMs is as high as possible.

### 6.2.1 Formulation of LM Adaptation as an MOP Problem

Let $P_{\mathbb{A}_n}$ denote an $n$-gram LM estimated on the application-specific data, $\mathbb{A}$, and $P_{\mathbb{S}_n}$ denote an $n$-gram LM estimated on the general domain data, $\mathbb{S}$. The KL divergence of the target $n$-gram distribution, $P_{\mathbb{T}}(\omega|h_\omega)$, from an estimated $n$-gram model, $P_n$, (which is either $P_{\mathbb{A}_n}$ or $P_{\mathbb{S}_n}$) is given by

$$D[P_{\mathbb{T}}||P_n] = \sum_h P_{\mathbb{T}}(h) \sum_\omega P_{\mathbb{T}}(\omega|h_\omega) \log \frac{P_{\mathbb{T}}(\omega|h_\omega)}{P_n(\omega|h_\omega)}. \tag{6.2.1}$$

As mentioned in Section 2.2.2.5, maximizing the likelihood function is equivalent to minimizing the KL divergence of the target model from a model obtained from the data (here $\mathbb{A}$), i.e., to minimizing $D[P_{\mathbb{T}}||P_{\mathbb{A}_n}]$. In the meantime, the conjugate prior density is the distribution which minimizes the KL divergence of the target posterior model from the prior distribution. Minimizing the KL divergence of the target model from the prior model makes the target model spread out as uniformly as possible without contradicting the given information. Thus, the use of conjugate prior density implicitly minimizes $D[P_{\mathbb{T}}||P_{\mathbb{S}_n}]$. Based on these two results, the LM adaptation problem solved by the SMAP method can be posed as the following MOP problem:

**Objective 1:** The target $n$-gram probabilities should be at a minimum distance from the background model. By minimizing the KL divergence of the target model from the background model, given new facts, the new distribution is being chosen which is as hard to discriminate from the well-trained background model as possible.

**Objective 2:** The target $n$-gram probabilities should be at a minimum distance from the distribution obtained from limited application-specific data. By minimizing the KL divergence of the target model from a model estimated from the application-specific data, the new distribution is suitable to describe the source generating the application-specific data.

#### *6.2.1.1 At-Least-Partially-Conflicting Objectives in LM Adaptation*

The KL divergence $D[P_{\mathbb{T}}||P_{\mathbb{A}_n}]$ (or $D[P_{\mathbb{T}}||P_{\mathbb{S}_n}]$) is minimal (and equal to zero) when the target model, $P_{\mathbb{T}}$, is exactly the same as $P_{\mathbb{A}_n}$ (or $P_{\mathbb{S}_n}$) and any deviation from $P_{\mathbb{A}_n}$ (or $P_{\mathbb{S}_n}$) results in a non-zero KL divergence. Because $P_{\mathbb{T}}$ cannot be exactly the same as $P_{\mathbb{S}_n}$ and $P_{\mathbb{A}_n}$ at the same time, minimizing $D[P_{\mathbb{T}}||P_{\mathbb{S}_n}]$ and minimizing $D[P_{\mathbb{T}}||P_{\mathbb{A}_n}]$ are at least partially conflicting objectives. Since the goal in LM adaptation is to compensate for the insufficiency of the application-specific data set by using the general domain corpus, the best approach to reliably estimating the $n$-gram probabilities is to establish a compromise between these two KL divergences.

Let $i$ denote an index from the index set $\{\mathbb{A}_1, \mathbb{A}_2, \mathbb{S}_1, \mathbb{S}_2\}$, where, $\mathbb{A}_n$ denotes the quantities of an $n$-gram model estimated from $\mathbb{A}$ and $\mathbb{S}_n$ denotes the quantities of an $n$-gram model estimated from $\mathbb{S}$. [1] Based on the results of the previous section, LM adaptation as solved by SMAP can be posed as an MOP problem in two different ways:

**A sequential optimization approach** It is possible to find the probabilities of unigrams, bigrams, and so on, one after another. We refer to this approach as *sequential ICO for language modeling and adaptation.* For a bigram model, this means solving the following optimization problems:

$$\begin{aligned} &\min \quad D[P_{\{\mathbb{T}_1}||P_{\{\mathbb{A}_1}] \\ &\min \quad D[P_{\{\mathbb{T}_1}||P_{\{\mathbb{S}_1}] \end{aligned} \tag{6.2.2}$$

and then

$$\begin{aligned} &\min \quad D[P_{\{\mathbb{T}_2}||P_{\{\mathbb{A}_2}] \\ &\min \quad D[P_{\{\mathbb{T}_2}||P_{\{\mathbb{S}_2}] \end{aligned} \tag{6.2.3}$$

**An all-in-once optimization approach** It is possible to compute the $n$-gram probabilities of all orders in only one optimization problem (instead of two). We refer to this approach as *$K$-objective ICO for language modeling and adaptation.* For a bigram model, this corresponds to solving the following multi-objective problem:

$$\min \quad D[P_{\{\mathbb{T}_1}||P_{\{\mathbb{A}_1}] \tag{6.2.4}$$

$$\min \quad D[P_{\{\mathbb{T}_1}||P_{\{\mathbb{S}_1}] \tag{6.2.5}$$

$$\begin{aligned} &\min \quad D[P_{\{\mathbb{T}_2}||P_{\{\mathbb{A}_2}] \\ &\min \quad D[P_{\{\mathbb{T}_2}||P_{\{\mathbb{S}_2}] \end{aligned}$$

### 6.2.2 Sequential ICO for Language Model Adaptation

Consider formulating the MOP formulation of the LM adaptation problem as a series of constrained optimization problems as in (6.2.2) and (6.2.3). In general, $P_{\mathbb{S}}$ is a considerably

[1] The divergences from unigram models as well as bigram models should be considered since backing-off is used when an unknown $n$-gram is observed during the recognition (test) stage.

larger model than $P_{\mathbb{A}}$, and hence one would expect the two objectives to have different scales. The two objectives in each of these problems can be rewritten so that so that we can expect similar "distances". This can be achieved by averaging each KL divergence by the number of $n$-grams each model has.

To solve the two-objective optimization problems in (6.2.2) and (6.2.3), two optimization subproblems need to be solved one after another. These two subproblems are

$$\begin{aligned} (\textsc{Problem 1}) \min_{P_{\mathbb{T}_n}(\omega|h_\omega)} \quad & \frac{1}{N_{n_{\mathbb{A}}}} D[P_{\mathbb{T}_n}(\omega|h_\omega)||P_{\mathbb{A}_n}(\omega|h_\omega)] \\ s.t. \quad & \frac{1}{N_{n_{\mathbb{S}}}} D[P_{\mathbb{T}_n}(\omega|h_\omega)||P_{\mathbb{S}_n}(\omega|h_\omega)] \leq d_{\mathbb{S}} \end{aligned}$$

$$\begin{aligned} (\textsc{Problem 2}) \min_{P_{\mathbb{T}_n}(\omega|h_\omega)} \quad & \frac{1}{N_{n_{\mathbb{S}}}} D[P_{\mathbb{T}_n}(\omega|h_\omega)||P_{\mathbb{S}_n}(\omega|h_\omega)] \\ s.t. \quad & \frac{1}{N_{n_{\mathbb{A}}}} D[P_{\mathbb{T}_n}(\omega|h_\omega)||P_{\mathbb{A}_n}(\omega|h_\omega)] \leq d_{\mathbb{A}} \end{aligned}$$

where $n = 1, 2$, and $d_{\mathbb{S}}$ and $d_{\mathbb{A}}$ are the constraint bounds obtained by perturbing the most recent averaged KL divergences. For solving (Problem 1), the constraint is incorporated into the optimization process using a Lagrangian multiplier $\lambda_S$. This results in the following Lagrangian function:

$$L(P_{\mathbb{T}_n}, \lambda_{S_n}) = D[P_{\mathbb{T}_n}||P_{\mathbb{A}_n}] + \lambda_{S_n} \cdot D[P_{\mathbb{T}_n}||P_{\mathbb{S}_n}] \tag{6.2.6}$$

By equating the gradient of this Lagrangian function with respect to $P_{\mathbb{T}_n}$ to 0 as

$$\left(1 + \log \frac{P_{\mathbb{T}_n}}{P_{\mathbb{A}_n}}\right) + \lambda_{S_n} \cdot \left(1 + \log \frac{P_{\mathbb{T}_n}}{P_{\mathbb{S}_n}}\right) = 0,$$

we obtain a closed form solution for the probabilities $P_{\mathbb{T}}(\omega|h_\omega)$ as

$$P_{\mathbb{T}}(\omega|h_\omega) = \frac{1}{Z(h_\omega)} [P_{\mathbb{A}}(\omega|h_\omega)]^{\frac{1}{1+\lambda_S}} [P_{\mathbb{S}}(\omega|h_\omega)]^{\frac{\lambda_S}{1+\lambda_S}} \tag{6.2.8}$$

where Lagrange multiplier $\lambda_S$ is the only unknown and $Z(h_\omega)$ is a history-dependent normalization factor. There is no closed-form solution for $\lambda$ but it can be found by iterative techniques. For this purpose, the constraint in (Problem 1) can be rewritten as

$$d(\lambda_S) = D^{\lambda_S}[P_{\mathbb{T}}(\omega|h)||P_{\mathbb{S}}(\omega|h)] - d_{\mathbb{S}} = 0 \tag{6.2.9}$$

The zeros of this function can be obtained using the Newton's method for nonlinear equations [80] as follows:

$$\lambda_{\mathbb{S},k+1} = \lambda_{\mathbb{S},k} - \frac{d(\lambda_{\mathbb{S},k})}{\frac{\partial d(\lambda_{\mathbb{S},k})}{\partial \lambda_{\mathbb{S},k}}} \tag{6.2.10}$$

The derivative of $d(\lambda_{\mathbb{S},k})$ with respect to $\lambda_{\mathbb{S},k}$ turns out to be a function of the target probabilities $P_T^{\lambda_{S,k}}(\omega|h)$ and is computed as:

$$\frac{\partial d^{\lambda_k}}{\partial \lambda_{\mathbb{S},k}} = \sum_h P_{\mathbb{T}}(h) \sum_\omega \left[1 + \log \frac{P_{\mathbb{T}}(\omega_h)}{P_{\mathbb{S}}(\omega|H)}\right] \cdot \left[\frac{1}{1+\lambda_{\mathbb{S},k}^2} \log \frac{P_{\mathbb{S}}}{P_{\mathbb{A}}} P_T^{\lambda_{\mathbb{S},k}}(\omega|h)\right]$$

The solution to (PROBLEM 2) is very similar: We just need to switch $P_{\mathbb{A}}$ and $P_{\mathbb{S}}$, and replace $\lambda_{\mathbb{S}}$ with $\lambda_{\mathbb{A}}$.

#### *6.2.2.1 Algorithmic Implementation*

The algorithmic implementation of the sequential ICO for language modeling and Adaptation is given in Table III. The algorithm starts with estimating the low-order $n$-gram probabilities for they are used as the history probabilities, $P_{\mathbb{T}}(h)$, for high-order $n$-grams. $D[P_{\mathbb{T}}(\omega|h)||P_{\mathbb{A}}(\omega|h)]$ is minimized by refining $\lambda_{S,k}$ so that $d(\lambda_{\mathbb{S},k})$ becomes roughly equal to 0 (see Equation (16)). Once such a $\lambda_{\mathbb{S}}$ is found, it is used to calculate the target $n$-gram probabilities as in Equation (15). After solving the first subproblem in this manner, the algorithm proceeds with solving the second subproblem. This iterative process stops when there is no progress in neither $D(P_{\mathbb{T}}||P_{\mathbb{A}})$ nor $D(P_{\mathbb{T}}||P_{\mathbb{S}})$. The relation $\lambda_{\mathbb{S}} = 1 \diagup \lambda_{\mathbb{A}}$ comes from the fact that to make the initial target probabilities of (PROBLEM 1) the same as those found after solving (PROBLEM 2), we should have $1 \diagup \lambda_{\mathbb{S}} = \lambda_{\mathbb{A}} \diagup (1 + \lambda_{\mathbb{A}})$.

### 6.2.3 $K$-Objective ICO for Language Model Adaptation

We choose to optimize each objective with constraints on the others. We then have four subproblems that will be solved one after another. For instance, one of these subproblems is

$$\begin{aligned} \min \quad & D[P||P_{A_2}] \\ \text{subject to} \quad & D[P||P_i] - d_i = 0, i \in \{\mathbb{A}_1, \mathbb{S}_1, \mathbb{S}_2\} \end{aligned}$$

**Table 2:** Sequential ICO algorithm for LM adaptation

I. Set $P_{\mathbb{T}}(h) = \frac{1}{N_1}$, where $N_1$ is the number of unigrams.

II. For n=1,2,...(i.e., unigrams, bigrams,...)

Repeat until no progress in $D(\cdot)$'s:

$\lambda_0 = 0.5$.

//Solve (PROBLEM 1):

ii. until $d(\lambda_{\mathbb{S},k}) \approx 0$.

Compute $\lambda_{\mathbb{S},k}$ from Equation (17).

iii. $\lambda_{\mathbb{S}} = \lambda_{\mathbb{S},k}$

iv. Compute $P_{\mathbb{T}}(\omega|h_\omega)$ from Equation (15).

v. $\lambda_{\mathbb{A}} = 1 \setminus \lambda_{\mathbb{S}}$.

//Solve (PROBLEM 2)

vi. until $d(\lambda_{\mathbb{A},k}) \approx 0$.

Compute $\lambda_{\mathbb{A},k}$ from Equation (17).

iii. $\lambda_{\mathbb{A}} = \lambda_{\mathbb{A},k}$

iv. Compute $P_{\mathbb{T}}(\omega|h_\omega)$ from Equation (15).

v. $\lambda_{\mathbb{S}} = 1 \setminus \lambda_{\mathbb{A}}$.

//New history probabilities for higher-order $n$-grams:

$P_{\mathbb{T}}(h_\omega) = P_{\mathbb{T}}(\omega|h_\omega)$.

For convenience, we restate this problem in an equivalent form as

$$\begin{aligned} \min \quad & \lambda_{A_2}(D[P||P_{A_2}] - d_{A_2}) \\ \text{subject to} \quad & D[P||P_i] - d_i = 0, i \in \{\mathbb{A}_1, \mathbb{S}_1, \mathbb{S}_2\} \end{aligned} \tag{6.2.12}$$

The reason for incorporating a scaling factor $\lambda_{A_2}$ and a reference value $d_{A_2}$ for the primary objective function (that is, the objective function of the problem in (6.2.12)) will shortly become clear. This problem can be solved by incorporating a Lagrange multiplier for each constraint. The resulting Lagrange function is

$$L(P, \lambda_i) = \sum_i \lambda_i (D[P||P_i] - d_i) \tag{6.2.13}$$

where $i \in \{\mathbb{A}_1, \mathbb{A}_2, \mathbb{S}_1, \mathbb{S}_2\}$. The target LM probabilities, $P_{\mathbb{T}}(\omega|h_\omega)$, are such that

$$\begin{aligned} \frac{\partial L}{\partial P_{\mathbb{T}}(\omega|h_\omega)} = \sum_i \lambda_i \frac{\partial D[P||P_i]}{\partial P_{\mathbb{T}}(\omega|h_\omega)} = \\ \sum_i \lambda_i \left(1 + \log \frac{P_{\mathbb{T}}(\omega|h_\omega)}{P_i(\omega|h_\omega)}\right) = 0 \end{aligned} \tag{6.2.14}$$

Solving this problem, we obtain a log-linear interpolation (LLI) of the component LMs as

$$P_{\mathbb{T}}(\omega|h_\omega) = \frac{1}{Z(h_\omega)} \prod_i P_i(\omega|h_\omega)^{\lambda_i} \tag{6.2.15}$$

where $Z(h_\omega)$ is a history-dependent normalization factor. Note that the same form of $P_{\mathbb{T}}(\omega|h_\omega)$ is obtained irrespective of the subproblem being solved. (Remember that there are four subproblems to be solved for a bigram target LM.) This is the reason for the inclusion of $\lambda_{A_2}$ and $d_{A_2}$ in (6.2.12).

#### *6.2.3.1 Algorithmic Implementation*

The algorithmic implementation of the proposed ICO method for LM adaptation is given in Table 1. The algorithm starts with some initial LM weights $\lambda$. These LM weights are refined in a manner that the subproblems in (6.2.12) are solved one after another.

In order to set some initial constraint bounds for each objective, all the KL divergences are evaluated. For the primary objective, the resulting KL divergence is discounted and set

**Table 3:** $K$-objective ICO algorithm for LM adaptation

I. Start with some initial LM weights, $\lambda_i, i \in \{A_1, S_1, A_2, S_2\}$.
II. Repeat

For each $i \in \{A_2, S_2, A_1, S_1\}$ (in this order):

i. Find the initial constraint bounds:

$d_k^0 = 0.9D[P_T||P_i]$, if $k = i$,

$d_k^0 = 1.1D[P_T||P_i]$, if $k \neq i$

ii. We want to solve:

$$\min \lambda_i(D[P||P_i] - d_i^0)$$

subject to $D[P||P_k] - d_k^0 = 0, k \neq i$

iii. Compute $\lambda_k, k \in \{A_2, S_2, A_1, S_1\}$ so that

$\mathcal{D}(\lambda_i)$ is minimized and $\mathcal{D}(\lambda_k), k \neq i$ is 0.

iv. Evaluate $P_T$ for these values of $\lambda_k$s.

as the constraint bound while for the other objectives, the KL divergences are inflated and set as the constraint bounds.

The LM weights are found so that (i) the deviation of the primary objective from the reference value is minimized, and (ii) there is no deviation of the objectives in the constraints from the constraint bounds.

#### *6.2.3.2 Finding the Component LM Weights*

There are no closed-form solutions for the Lagrange multipliers, $\lambda_i$, but they can be found by iterative techniques. Given the constraint bounds $d_i$, $\lambda_i$'s should be such that the derivative of the Lagrange function with respect to $\lambda_i$ vanish (by KKT optimality conditions), i.e.,

$\frac{\partial L}{\partial \lambda_i} = D[P||P_i] - d_i = 0, \forall i.$

Let $\mathcal{D}(\lambda_i)$ denote the deviation of a KL divergence from the respective constraint bound, i.e., $\mathcal{D}(\lambda_i) = D[P||P_i] - d_i$. It is a function of $\lambda_i$ since $P(\omega|h_\omega)$ used for calculating $D[P||P_i]$ is obtained by (6.2.15). The zeros of this function can be obtained using the Newton's method for nonlinear equations as $\lambda_{i+1} = \lambda_i - \mathcal{D}(\lambda_i)/\mathcal{D}'(\lambda_i)$ where

$$\begin{aligned} \mathcal{D}'(\lambda_i) &= \frac{\partial \mathcal{D}}{\partial \lambda_i} = \frac{\partial \mathcal{D}}{\partial P_T} \cdot \frac{\partial P_T}{\partial \lambda_i} \\ &= \sum_{(\omega, h_\omega)} \left[ P_{\mathbb{T}}(h_\omega)(1 + \log \frac{P_{\mathbb{T}}(\omega|h_\omega)}{P_i(\omega|h_\omega)}) \right] \cdot \left[ (1 - \frac{\lambda_i}{(\sum_j \lambda_j)^2})(\log P_i(\omega|h_\omega)) P_{\mathbb{T}}(\omega|h_\omega) \right]. \end{aligned} \tag{6.2.16}$$

### 6.2.4 Adjusting the Constraint Bounds

If it was possible to know the KL divergences that a desirable target LM would yield, this problem could be relatively straightforward. However, since we initially are not given these KL divergences, we choose to adjust the goals in an iterative manner.

The constraint bounds are obtained by perturbing the most recent KL divergences. The reference value for the primary objective is obtained by decreasing the most recent value (for instance, by multiplying by 0.9). The constraint bounds for the objectives in the constraints are obtained by slightly increasing their most recent values (for instance, by multiplying by 1.1). This is to ensure that the KL divergence of the target model from the primary objective is reduced while the others are tolerated to increase. The algorithmic implementation of the proposed ICO method for LM adaptation is given in Table 1.

#### *6.2.4.1 Finding and Validating the Step-Size*

The two objectives in the LM adaptation application interact in a complicated manner. It is hard to analyze which objective has what kind of impact on the speech recognition performance, For this reason, the perplexity is used as the utility function to judge the preferability of the new step found with the Armijo rule. That is, with the ICO method for the LM adaptation application, the step-size found by Armijo rule is validated and the LM weights are updated *only* in those cases which do not result in degradation in terms of perplexity.

## *6.3 Experimental Results*

In our ASR experiments, the training speech material was the SI-84 set with 7077 utterances from 84 speakers, which was roughly equivalent to 15.3 hours of speech. The test set was the November'92 evaluation set with 330 utterances from 8 speakers. We used the 5K-word vocabulary. A gender-independent (GI) ASR baseline system with tied-state cross-word triphone models was built using HTK [30]. The phone set was consisting of 45 phones and the acoustic vectors were composed of 12 MFCCs, log energy, velocity, and acceleration coefficients. The HMM parameters were trained using classical maximum likelihood estimation. Each HMM had three states with eight Gaussian mixture components per state. We found the WER to be 5.03% using the already-provided benchmark trigram LM.

### 6.3.1 Experimental Results with SMAP

In this section, we explore the performance of SMAP in ASR experiments. Since the WSJ corpus do not come with a separate adaptation set, we construct a fake adaptation set to investigate the use of SMAP for LM adaptation.

#### *6.3.1.1 Experiments to Investigate the Use of SMAP*

We first attempted to see if the SMAP LM training method is useful in comparison to the baseline ML-based training with Katz backoff and Good-Turing smoothing. For this purpose, we used the entire WSJ0 text material, which is denoted as $\mathbb{O}$, both to estimate a prior distribution and to compute the likelihood of the text. We set $\epsilon = 0.0001$ and $\rho = 0.01$. The resulting LM gave a WER of 5.01% while the baseline ML-based LM gave a WER of 5.03%. This slight improvement is a result of the use of the hierarchical prior that is propagated through the $n$-gram trees. To see if SMAP could help reduce WER, we used the test text as adaptation set and found that the WER was reduced from 5.03% to 1.73% (using the same acoustic models). This 66% relative reduction in WER motivated us to explore the issue of designing a useful adaptation set.

One way to have an effective LM adaptation set is to have such a set, $\mathbb{A}$, that matches the statistics of the testing set, T, as much as possible. Starting with the original training

set, $\mathbb{O}$, we investigated different options to split $\mathbb{O}$ into two subsets, one, denoted as $\mathbb{S}$, for training background language models and the other, denoted as $\mathbb{A}$, for application-specific LM adaptation.

#### *6.3.1.2 Performance of ML Models with an Irrelevant Adaptation Set*

As a first attempt to construct an application-specific adaptation set, we randomly selected 1 out-of-every 10 sentences from the original training set, $\mathbb{O}$. As it will turn out, doing so resulted in a set which was totally irrelevant to the test domain. We denote this set with a subscript $i$ (as $\mathbb{A}_i$) to emphasize that it was irrelevant to the test domain. The remaining sentences, which composed 90% of $\mathbb{O}$, were used as the general domain data. We denote this set as $\mathbb{S}_i = \mathbb{O} - \mathbb{A}_i$[2].

Some key experimental results with this setup are reported in Table II. As shown there, the WER was 5.04% when only $\mathbb{S}_i$ was used for ML training of a language model. Note that $\mathbb{S}_i$ represents 90% of $\mathbb{O}$ and the WER is almost the same as the WER of the baseline system. When one-fifth of $\mathbb{A}_i$ was used, the WER was 9.12% and it reduced to 7.23% when the entire $\mathbb{A}_i$ was used for ML training.

We then trained ML models using $\mathbb{S}_i$ *plus* some portions of $\mathbb{A}_i$. The corresponding WERs are shown in the rightmost column of Table II. It is observed that whether one-fifth or the entire $\mathbb{A}_i$ was used together with $\mathbb{S}_i$, the WER did not change significantly from the WER obtained with $\mathbb{S}_i$ only.

These experimental results indicated that $\mathbb{A}_i$ has similar characteristics to $\mathbb{S}_i$ and hence was not *informative* about the test domain. Despite its weakness to provide useful information about the test domain and hence to reduce the WER, experiencing with it led us to construct a suitable adaptation set. This is discussed next.

[2]All the constructed sets are padded by additional (2 to 8) sentences so that there is no word that the test data set has but not in the training/adaptation data. It is then guaranteed that the gain obtained by using difference sets is not due to out-of-vocabulary (OOV) effect.

**Table 4:** WERs when ML models are trained using the irrelevant adaptation and training sets.

| Data set | Size (%) | WER (%) | Data set | Size (%) | WER (%) |
|---|---|---|---|---|---|
| $\mathbb{S}_i$ | 90 | 5.04 | | | |
| $\frac{1}{5}\mathbb{A}_i$ | 2 | 9.12 | $\mathbb{S}_i + \frac{1}{5}\mathbb{A}_i$ | 92 | 5.01 |
| $\frac{2}{5}\mathbb{A}_i$ | 4 | 8.41 | $\mathbb{S}_i + \frac{2}{5}\mathbb{A}_i$ | 94 | 5.03 |
| $\frac{3}{5}\mathbb{A}_i$ | 6 | 7.55 | $\mathbb{S}_i + \frac{3}{5}\mathbb{A}_i$ | 96 | 5.04 |
| $\frac{4}{5}\mathbb{A}_i$ | 8 | 7.27 | $\mathbb{S}_i + \frac{4}{5}\mathbb{A}_i$ | 98 | 5.06 |
| $\mathbb{A}_i$ | 10 | 7.23 | $\mathbb{S}_i + \mathbb{A}_i$ | 100 | 5.03 |

#### *6.3.1.3 An Artificial Relevant Adaptation Set*

We employed a simple but effective technique for constructing a relevant application-specific adaptation set, denoted as $\mathbb{A}_r$ (with a subscript $r$ to emphasize that it is a relevant adaptation set). Let the set $\mathbb{O} - \mathbb{A}_r$ serve as a general domain data set and be denoted as $\mathbb{S}_r$. We first formed an initial general domain data set, which we denote as $\mathbb{S}_r^0$ (with a superscript 0 to emphasize that it was an initialization). $\mathbb{S}_r^0$ was constructed by randomly selecting some of the sentences in $\mathbb{O}$, in our experiments 1 out-of-every 5 sentences. The sentences in $\mathbb{O} - \mathbb{S}_r^0$ were to be either selected as adaptation data sentences or added to $\mathbb{S}_r^0$. This process continued such that $\mathbb{A}_r$ includes those sentences of $\mathbb{O} - \mathbb{S}_r^0$ that contained a test $n$-gram that $\mathbb{S}_r^0$ did not have. The set of such sentences was further enriched by adding random sentences from $\mathbb{O}$ so that $\mathbb{A}_r$ has the same size as $\mathbb{A}_i$. The data that were not selected for $\mathbb{A}_r$ were used as the training set, which is denoted as $\mathbb{S}_r$.

#### *6.3.1.4 Comparison of $\mathbb{A}_i$ and $\mathbb{A}_r$*

Some crucial statistics about the two sets, $\mathbb{A}_i$ and $\mathbb{A}_r$ are reported in Tables III and IV, where $\mathbb{A}_i'$ and $\mathbb{A}_r'$ denote some portion of $\mathbb{A}_i$ and $\mathbb{A}_r$, respectively. A comparison of Tables III and IV revealed several features of the two adaptation sets. We discuss them next.

First of all, the total number of events in the same-size subsets of $\mathbb{A}_i$ and $\mathbb{A}_r$ were very close, as shown in the third columns from the left of Tables III and IV. Similarly, the total counts of events in the general domain data set *plus* the available adaptation data set ($\mathbb{S}_i + \mathbb{A}_i'$ or $\mathbb{S}_r + \mathbb{A}_r'$) were also very similar, as shown in the fourth columns of Tables III

**Table 5:** Data statistics for $\mathbb{A}_i$ and $\mathbb{S}_i$.

| | | Total Events in the Set | Total Events in $\mathbb{S}_i + \mathbb{A}'$ | Test Events Unseen in $\mathbb{S}_i + \mathbb{A}'$ |
|---|---|---|---|---|
| $\mathbb{S}_i$ | $c_1$ | 4,985 | - | 0 |
| | $c_2$ | 1,556,850 | | 144 |
| | $c_3$ | 8,519,787 | | 886 |
| $\mathbb{A}' = \frac{2}{5}\mathbb{A}_i$ | $c_1$ | 4,922 | 4,985 | 0 |
| | $c_2$ | 274,344 | 1,590,815 | 142 |
| | $c_3$ | 796,365 | 8,753,573 | 873 |
| $\mathbb{A}' = \mathbb{A}_i$ | $c_2$ | 4,962 | 4,985 | 0 |
| | $c_2$ | 475,065 | 1,639,687 | 137 |
| | $c_3$ | 1,652,796 | 9,181,362 | 854 |

and IV. Both of these two observations are important because only then it can be claimed that any reduction in WER is because of the characteristics rather than the size of the adaptation set.

As shown in the rightmost columns of Tables III and IV, the numbers of test $n$-grams that remains unseen in $\mathbb{S}_i + \mathbb{A}'_i$ and $\mathbb{S}_r + \mathbb{A}'_r$ were substantially different. By comparing the second and the bottom rows, we find that $\mathbb{A}_i$ provides only 7 new test bigrams (number of unseen bigrams reduced from 144 to 137) and 32 new test trigrams (number of unseen trigrams reduced from 886 to 854) to $\mathbb{S}_i$, whereas $\mathbb{A}_r$ provides 166 new test bigrams (number of unseen bigrams reduced from 303 to 137) and 510 new test trigrams (number of unseen trigrams reduced from 1364 to 854) to $\mathbb{S}_r$.

As shown in the third row of the last column of Table III, when $\frac{2}{5}\mathbb{A}_i$ was used, only 2 new test bigrams (number of unseen bigrams reduced from 144 to 142) and 13 new test trigrams (number of unseen trigrams reduced reduced from 886 to 873) were added. On the other hand, as shown in the third row of the last column of Table IV, when $\frac{2}{5}\mathbb{A}_r$ was used, 119 new test bigrams number of unseen bigrams reduced reduced from 303 to 184) and 310 new test trigrams (number of unseen trigrams reduced from 1,364 to 1,054)were added. The conclusion is that $\mathbb{A}_r$ provides $\mathbb{S}_r$ with *new specific* information about the test domain, while $\mathbb{A}_i$ did not give as much new information.

**Table 6:** Data statistics for $\mathbb{A}_r$ and $\mathbb{S}_r$.

| | | Total Events in the Set | Total Events in $\mathbb{S}_r + \mathbb{A}'$ | Test Events Unseen in $\mathbb{S}_r + \mathbb{A}'$ |
|---|---|---|---|---|
| $\mathbb{S}_r$ | $c_1$ | 4,985 | - | 0 |
| | $c_2$ | 1,556,766 | | 303 |
| | $c_3$ | 8,519,647 | | 1,364 |
| $\mathbb{A}' = \frac{2}{5}\mathbb{A}_r$ | $c_1$ | 4,919 | 4,985 | 0 |
| | $c_2$ | 274,254 | 1,590,617 | 184 |
| | $c_3$ | 796,269 | 8,753,100 | 1,054 |
| $\mathbb{A}' = \mathbb{A}_r$ | $c_1$ | 4,960 | 4,985 | 0 |
| | $c_2$ | 474,444 | 1,639,687 | 137 |
| | $c_3$ | 1,653,349 | 9,181,362 | 854 |

#### *6.3.1.5 Performance of ML Models with $\mathbb{A}_r$*

Similar to those shown in Table I, the WER results in Table V were obtained when $\mathbb{A}_r$ and $\mathbb{S}_r$ were used to train ML-based models. The WER was 6.63% when $\mathbb{S}_r$ was used, while it was 7.86% when one fifth of $\mathbb{A}_r$ was used. As more of $\mathbb{A}_r$ was used, the WER gradually reduced to 5.17%. As a reminder the WER was 7.23% when the entire $\mathbb{A}_i$ was used. This difference is as expected since $\mathbb{A}_r$ is richer than $\mathbb{A}_i$ in terms of the test $n$-grams much-needed for adaptation to the test domain. In addition, $\mathbb{A}_r$ gives lower WER than $\mathbb{S}_r$, which shows that $\mathbb{A}_r$ was limited yet more relevant to the test domain than $\mathbb{S}_r$ (while this was not valid when $\mathbb{A}_i$ and $\mathbb{S}_i$ were used). Furthermore, using even only three-fifth of $\mathbb{A}_r$, which is only 6% of $\mathbb{O}$, yielded a lower WER than $\mathbb{S}_r$. As expected, a low WER is achievable with a reasonably small set if it is a relevant set.

As shown in the rightmost column of Table V, there was a significant improvement in WER when more of $\mathbb{A}_r$ was used in addition to $\mathbb{S}_r$ to train ML models. The WER was 6.35% with the addition of only one-fifth of $\mathbb{A}_r$, which was better than using only $\mathbb{S}_r$. As more of $\mathbb{A}_r$ was included in training, the WER gradually reduced from 6.35% to 5.03%. As a reminder, when $\mathbb{A}_i$ was incrementally added to build an LM, the ASR performance basically did not change.

**Table 7:** WERs when ML models are trained using the relevant adaptation and general domain datasets.

| Data set | Size (%) | WER (%) | Data set | Size (%) | WER (%) |
|---|---|---|---|---|---|
| $\mathbb{S}_r$ | 90 | 6.63 | | | |
| $\frac{1}{5}\mathbb{A}_r$ | 2 | 7.86 | $\mathbb{S}_r + \frac{1}{5}\mathbb{A}_r$ | 92 | 6.35 |
| $\frac{2}{5}\mathbb{A}_r$ | 4 | 6.84 | $\mathbb{S}_r + \frac{2}{5}\mathbb{A}_r$ | 94 | 5.98 |
| $\frac{3}{5}\mathbb{A}_r$ | 6 | 5.90 | $\mathbb{S}_r + \frac{3}{5}\mathbb{A}_r$ | 96 | 5.79 |
| $\frac{4}{5}\mathbb{A}_r$ | 8 | 5.42 | $\mathbb{S}_r + \frac{4}{5}\mathbb{A}_r$ | 98 | 5.40 |
| $\mathbb{A}_r$ | 10 | 5.17 | $\mathbb{S}_r + \mathbb{A}_r$ | 100% | 5.03 |

These experimental results verified that $\mathbb{A}_r$ provides new information about the application domain that the training data did not have. Therefore, we used it to illustrate the SMAP LM adaptation experimentally.

#### *6.3.1.6 Performance of SMAP LM Adaptation*

We performed experiments to find the effect of different parameters, $\rho$ and $\epsilon$, on the perplexity and WER.

*1. Effect of $\rho$*

First, we explored the effect of the forgetting factor, $\rho$. It is a parameter serving two purposes: First, it controls how much information each node inherits from its parent node (see Eq. (13)). Second, it controls how much the prior information contributes in SMAP probability calculation (see Eq. (16)).

The perplexity and WER are plotted as functions of $\rho$ for different values of $\epsilon$ in Fig. 2 and Fig. 3, respectively. Fig. 2 shows that the perplexity first reduced as $\rho$ increased, reaching its minimum at 47.65 at $\rho = 0.01$ for $\epsilon = 0.01$, and increased thereafter. Upon a comparison of Fig. 2 and Fig. 3, we observe that the perplexity and WER in general change in parallel except that the perplexity was higher when $\rho = 0.1$ than when $\rho = 10^{-6}$, yet the WER was lower when $\rho = 0.1$ than when $\rho = 10^{-6}$.

This behavior suggests, first, that when the contributions from the parent nodes were very small or very large, i.e., $\rho$ was at extremes, the resulting LMs were not so right. When $\rho$ was large, the node-specific information is dominated by the parent-specific information.

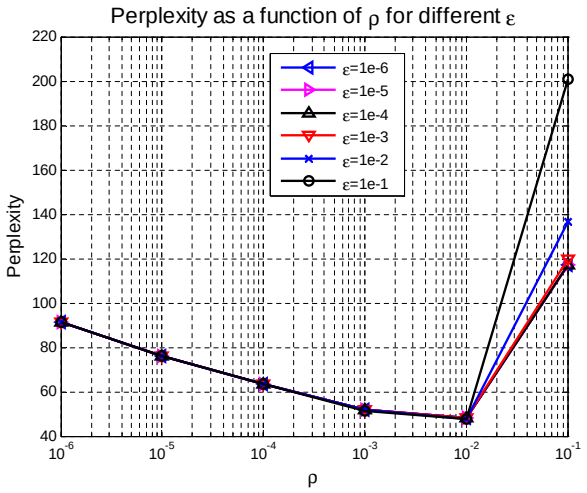



**Figure 13:** The perplexity when $\rho$ is changed for fixed $\epsilon$.

When $\rho$ was small, the nodes did not inherit significant information from parent nodes. Secondly, with an appropriate selection of $\rho$, the prior information helped improve the performance by contributing in the $n$-gram probability estimation (Eq. (13)).

*2. Effect of $\epsilon$*

Secondly, we explored the effect of changing $\epsilon$ by fixing $\rho$. The parameter $\epsilon$ has an influence on the estimation of hyperparameters at the root nodes, which are then propagated to all other tree nodes. The perplexity and WER are plotted as functions of $\epsilon$ for different values of $\rho$ in Fig. 4 and Fig. 5, respectively. Note that Fig. 4 and Fig. 5 are just other ways of looking at Fig. 2 and Fig. 3.

We observe that the perplexity slightly reduced as $\epsilon$ increased except for the case when $\rho = 0.1$. The same result held true for the change of WER as well. When $\epsilon$ is very small, the $n$-gram counts in the root nodes are excessively smoothed. This means that even when the frequencies of two unigrams differed by orders of magnitude, the corresponding hyperparameters were very close.

*3. Effect of the Size of the Adaptation Data Set*

After the above and further experiments, we found that the minimum of the perplexity was achieved when $\rho = 0.005$ and $\epsilon = 0.2$, while the minimum of the WER was achieved when $\rho = 0.01$ and $\epsilon = 0.1$. In Table VI, we report the performance of the SMAP adaptation with these two settings when different amounts of adaptation data were used. The second

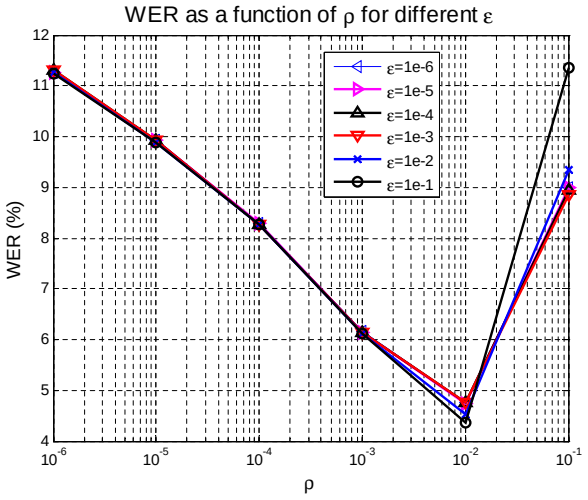



**Figure 14:** The WER when $\rho$ is changed for fixed $\epsilon$.

column from the left of Table VI denotes the frequency of those test set trigrams that the given set includes, which is referred to as the percentage of trigram hits.

When an LM was adapted using larger subsets of $\mathbb{A}_r$, the trigram hits increased from 76.16% to 82.24% and the perplexity reduced from 67.28 to 45.48. As would be anticipated from these two changes, the WER reduced from 6.11% to 4.91%.

Similar experiments were repeated with $\rho = 0.01$ and $\epsilon = 0.1$, which was the combination of parameters that yielded the best WER. The results of these experiments are also reported in Table VI. Although the perplexity results in the second experiment were higher than those in the first experiments, the WERs were lower in the second experiment when the size of the adaptation set was large enough, i.e., $\geq 80\%$ of $\mathbb{A}_r$.

*4. SMAP Adaptation with $\mathbb{A}_i$ and $\mathbb{A}_r$*

One further experimental study concerned about comparing the performance of the SMAP LM adaptation framework using $\mathbb{A}_i$ and $\mathbb{A}_r$. Our experimental findings are plotted in Fig. 6. The curves marked with "□" and "○" were obtained when $\mathbb{A}_i$ and $\mathbb{A}_r$ were used, respectively.

Several observations can be drawn from Fig. 6. First of all, using $\mathbb{A}_r$ reduced the WER considerably. SMAP LM adaptation framework was able to reduce WER to 4.37% using the entire $\mathbb{A}_r$, which was relatively 15.5% better than the WER obtained with $\mathbb{A}_r$. On the

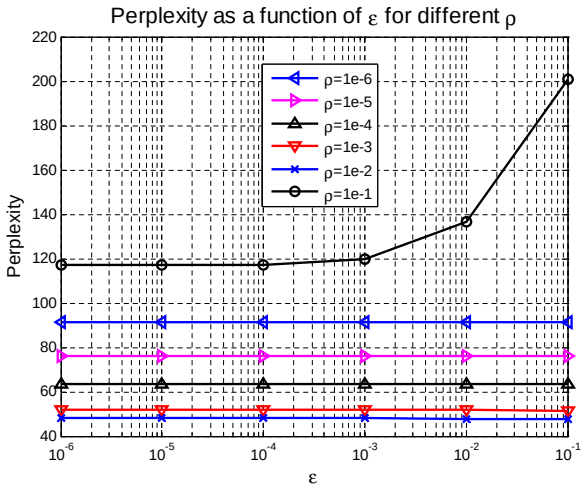



**Figure 15:** The perplexity is slightly changed with $\epsilon$ for reasonable values of $\rho$.

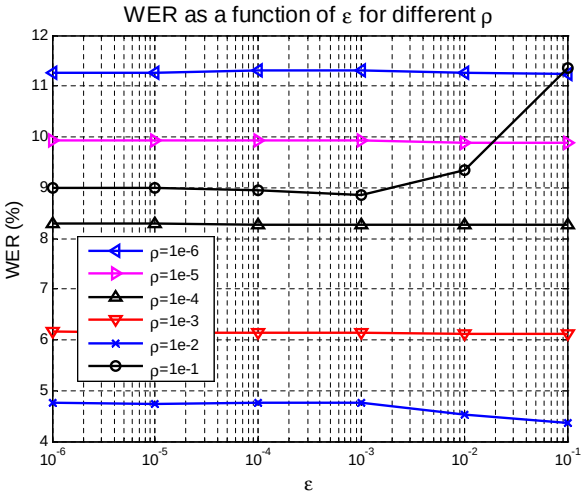



**Figure 16:** The WER is slightly changed with $\epsilon$ for reasonable values of $\rho$.

**Table 8:** Perplexity and WER with two settings of $\rho$ and $\epsilon$.

| | | $\rho = 0.005, \epsilon = 0.2$ | | $\rho = 0.01, \epsilon = 0.1$ | |
|---|---|---|---|---|---|
| Data set | Trigram hits | PP | WER (%) | PP | WER (%) |
| $\mathbb{S}_r$ | 71.69 | 58.13 | 6.63 | 58.13 | 6.63 |
| $\mathbb{S}_r + \frac{1}{5}\mathbb{A}_r$ | 76.16 | 67.28 | 6.11 | 76.71 | 6.58 |
| $\mathbb{S}_r + \frac{2}{5}\mathbb{A}_r$ | 78.29 | 59.64 | 5.75 | 65.08 | 5.96 |
| $\mathbb{S}_r + \frac{3}{5}\mathbb{A}_r$ | 80.32 | 52.15 | 5.55 | 55.92 | 5.29 |
| $\mathbb{S}_r + \frac{4}{5}\mathbb{A}_r$ | 81.39 | 48.77 | 5.17 | 51.65 | 4.89 |
| $\mathbb{S}_r + \mathbb{A}_r$ | 82.24 | 45.48 | 4.91 | 47.84 | 4.37 |

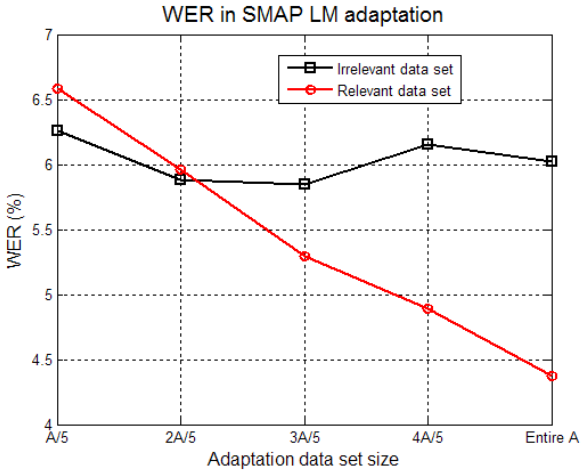



**Figure 17:** The performance of SMAP LM adaptation when relevant adaptation data is made available.

other hand, performing SMAP adaptation with $\mathbb{A}_i$ proved itself useless. This is because not only the ML modeling yielded better WER results than LM adaptation but also using more of $\mathbb{A}_i$ did not yield any improvement.

### 6.3.2 Experimental Results with ICO

In this section, we report our experimental results comparing the proposed LM adaptation approach with log-linear interpolation and SMAP.

#### *6.3.2.1 Performance Evaluation of Sequential ICO for Language Modeling and Adaptation*

The average KL divergences of the target model from two independently models, one trained on the general domain data and the other trained on the application-specific data, are being minimized. The goal in ICO is then to find a tradeoff between these two distance measures. Our experimental result on the change of the KL divergences is shown in Figure 6.3.2.1. The constraints were obtained so that the most recent objective functions were increased by 1%. As shown in Figure 6.3.2.1, each objective followed a zigzag pattern throughout the iterative process. This is because the two objectives attempt to modify the target model probabilities to be close to their respective model.

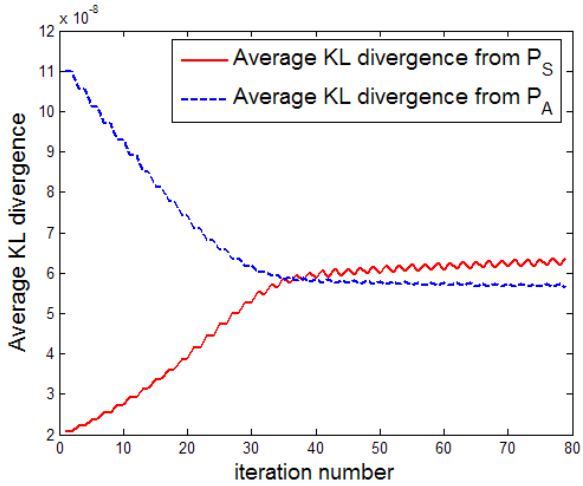



**Figure 18:** The change of the average KL divergences of the target model to the background model $P_{\mathbb{S}}$ and to the application-specific model $P_{\mathbb{A}}$.

#### *6.3.2.2 Performance Evaluation of K-objective ICO for Language Modeling and Adaptation*

We then performed experiments on model perplexity and ASR word error rate (WER) to compare the performance of the proposed LM adaptation framework with the SMAP method and Klakow's LLI model. Our experimental results are reported in Table 6.3.2.2. In our ASR experiments, we used the same design as in [109]. The SMAP model was trained with $\rho = 0.0001$ for the propagation of hyper-parameters, $\epsilon = 0.01$, and $\rho = 0.1$ for the estimation of MAP probabilities.

As shown in Table 6.3.2.2, the MOP-based approach is superior to SMAP by 3.8% in terms of relative reduction in WER in ASR experiments. This is because MOP leaves more flexibility in finding the $n$-gram estimates while SMAP attempts to merge the conflicting goals a priori in an overall function. In the meantime, in comparison to Klakow's LLI, the MOP solution performs relatively 2.1 % better in terms of WER. Although both had the same form of the solution, the distinction was in the way the LM weights (i.e., the Lagrange multipliers) were estimated. Although the improvements do not seem significant, the major gains in solving the LM adaptation problem with MOP are twofold: (i) Upon observing the behavior of each objective, we have full freedom to tune the system into different operating points to meet different requirements. (ii) Meantime, by observing each objective, we can

**Table 9:** Comparison of LM adaptation methods of interest

| | Bigram | | Trigram | |
|---|---|---|---|---|
| Model | PP | WER (%) | PP | WER (%) |
| LLI | 80.06 | 7.06 | *(TBC)* | *(TBC)* |
| SMAP | 80.53 | 7.19 | 47.84 | 4.37 |
| MOP | 79.18 | 6.91 | *(TBC)* | *(TBC)* |

easily avoid extremes, i.e., the cases that the target LM is too dependent on the application specific data or on the general domain data.

# CHAPTER VII

# SPOKEN LANGUAGE IDENTIFICATION

In this chapter, we develop an SOP-based and an MOP-based approach for the problem of automatic spoken language identification, where the standard detection performance evaluation measures *false-rejection* (or miss) and *false-acceptance* (or false alarm) rates for a number of languages are to be simultaneously minimized. LID systems might be used as a pre-processing stage for understanding systems and for human listeners, and find applications in, for example, a hotel lobby or an international airport where one might speak to a multi-lingual voice-controlled travel information retrieval system.

## *7.1 An SOP-Based Baseline*

As part of our submission to NIST LRE organized in 2005, we designed a set of one-versus-rest type linear classifiers to determine whether or not the speech is from a given target language. This design was based on the optimization of an overall training objective and was the reference system for our future MOP-based design. We discuss a detailed description of this system in the this section, and experimentally compare it to MOP-based design in Section 7.3.

In our design, we first noted that the system performance is to be evaluated by an overall detection cost function, which is roughly the average error rate, $E^{avg}$,

$$E^{avg}(\Theta) = \frac{1}{2M} \sum_{m=1}^{M} [E_m^{FR}(\Theta) + E_m^{FA}(\Theta)], \tag{7.1.1}$$

where $E_m^{FA}(\Theta)$ and $E_m^{FR}(\Theta)$ are the false-acceptance (FA) (or equivalently, false-alarm) rate and the false-rejection (FR) (or equivalently, miss) rate of the $m^{th}$ language, respectively, and $\Theta$ denotes the collection of the unknown classifier parameters. This overall performance function, which combines all the design objectives, can be optimized if it can be approximated as a function of the classifier parameters.

### 7.1.1 Formulation of LID as an SOP Problem

To obtain an approximation of the empirical FR and FA rates, the minimum classification error rate classification (MCE) framework is followed using linear discriminant functions (LDFs). As a reminder, for LDFs given as

$$g(\mathbf{x}, \mathbf{w}_m) = \mathbf{w}_m^T \mathbf{x}, \mathbf{w}_m \in \Re^n, m = 1, ..., M,$$

the MCE method defines a loss function as

$$\ell_m(\mathbf{x}, \Theta) = \frac{1}{1 + \exp(-\alpha \pi_m(\mathbf{x}, \Theta) + \beta_m)}$$

where $\Theta$ is the set of all unknown LDF weight vectors, $\mathbf{w}_m \in \Re^n$, (i.e., $\Theta = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$) and

$$\pi_m(\mathbf{x}, \Theta) = -g(\mathbf{x}, \mathbf{w}_m) + \log[\frac{1}{M-1} \sum_{i \neq m} \exp(\eta g(\mathbf{x}, \mathbf{w}_i))]^{\frac{1}{\eta}}.$$

When $\pi_m(\mathbf{x}, \Theta)$ is very small, which implies correct classification, virtually no loss is incurred. When $\pi_m(\mathbf{x}, \Theta)$ is very large, it leads to a penalty which essentially becomes a classification/recognition error count. Hence, using $\ell(\cdot)$, we can approximate the FR and FA rates for the $m^{th}$ category as functions of the classifier parameters $\Theta$ as

$$E_m^{FR}(\Theta) \approx \frac{1}{|\mathbb{C}_m|} \sum_{\mathbf{x} \in \mathbb{C}_m} [1 - \ell_m(\mathbf{x}, \Theta)], \tag{7.1.2}$$

$$E_m^{FA}(\Theta) \approx \frac{1}{|\mathbb{S}| - |\mathbb{C}_m|} \sum_{\mathbf{x} \notin C_m} \ell_m(\mathbf{x}, \Theta), \tag{7.1.3}$$

where $\mathbb{S}$ is the set of training samples and $|.|$ denotes the size of a set. We can find those parameters for which the average of all the FA and FR rates is minimum using the steepest descent algorithm, i.e., we update the classifier parameters as

$$\mathbf{w}_m^{(t+1)} = \mathbf{w}_m^{(t)} - \epsilon^{(t)} \sum_1^M [\frac{\partial E_m^{FR}(\Theta)}{\partial \mathbf{w}_m} + \frac{\partial E_m^{FR}(\Theta)}{\partial \mathbf{w}_m}]$$

where $\mathbf{w}_m^{(t)}$ is the vector of LDF weights of the $m^{th}$ language after the $t^{th}$ iteration of the steepest descent algorithm, and $\epsilon$ is a preset step size.

## *7.2 An MOP-based Approach: ICO for LID*

When the LID task is defined as *detecting* the language spoken in a speech utterance, the actual goal is to minimize the FA and FR rates for each of the target languages rather than to minimize their average.

### 7.2.1 Formulation of LID as an MOP Problem

We can formulate the LID problem as an MOP problem with a total of $(2 \cdot M)$ objectives, where each individual objective is either an FA or an FR rate for a target language/dialect. Thus, the LID problem involves the following objectives:

$M$ **FR Objectives:** The probability of classifying the samples of a target language as that of another language should be as small as possible for each of the $M$ languages.

$M$ **FA Objectives:** The probability of classifying the samples of some language as that of a target language should be as small as possible for each of the $M$ languages.

Mathematically, the optimization of these $(2 \cdot M)$ objectives corresponds to solving the following MOP problem:

$$\text{(MOP Problem)} \qquad \begin{aligned} &\min E_m^{FR}(\Theta), m = 1, ..., M \\ &\min E_m^{FA}(\Theta), m = 1, ..., M. \end{aligned}$$

In the SOP-based approach of the previous section, it is implicitly assumed that the individual error rates will be as small as possible when the average error rate was at its minimum. This implicit assumption may result in significant degradation in some objectives. On the other hand, the MOP-based approach under discussion *directly* attempts to find how to make individual error rates as small as possible without significant degradation in any one of them.

### 7.2.2 MOP-based LID using ICO

Consider formulating the MOP problem as a set of $2 \cdot M$ subproblems in the form of

$$\begin{aligned} (\text{Problem 1}) \quad \min_{\Theta} \quad & E_k^{FR}(\Theta) \\ \text{subject to} \quad & E_p^{FR}(\Theta) \leq e_p,\ p = 1, ..., M, p \neq k, \\ & E_r^{FA}(\Theta) \leq e_r,\ r = 1, ..., M, \end{aligned} \tag{7.2.1}$$

$$\begin{aligned} (\text{Problem 2}) \quad \min_{\Theta} \quad & E_k^{FA}(\Theta) \\ \text{subject to} \quad & E_p^{FA}(\Theta) \leq e_p,\ p = 1, ..., M, p \neq k, \\ & E_r^{FR}(\Theta) \leq e_r,\ r = 1, ..., M, \end{aligned} \tag{7.2.2}$$

for all $k = 1, ..., M$, where $e_i$'s are the upper bounds for the objective functions that are to be attained.

#### *7.2.2.1 Solution Methodology*

For simplicity, let $E_k(\Theta)$ denote the error rate being minimized and $E_p(\Theta)$ denote the error rates embedded into the constraints. Then, the following subproblem is a common form of the subproblems in (7.2.1) and (7.2.2):

$$\begin{aligned} \min_{\Theta} \quad & E_k(\Theta) \\ \text{subject to} \quad & E_p(\Theta) \leq e_p,\ p = 1, ..., 2 \cdot M, p \neq k. \end{aligned} \tag{7.2.3}$$

The augmented Lagrangian function for the constrained optimization problems in (7.2.3) is

$$L_A^k(\Theta, \lambda, \rho) = E_k(\Theta) + \sum_{p \neq k} \psi_p(\Theta, \lambda_p, \rho_p),$$

where $\lambda_p > 0$ and $\rho_p \geq 0$ are the Lagrange multiplier and penalty parameter, respectively, associated with the $p^{th}$ constraint. The quantities $\psi_p(\Theta, \lambda, \rho)$ are given by:

$$\psi_p(\Theta, \lambda_p, \rho_p) = \begin{cases} -\lambda_p(e_p - E_p(\Theta)) + \frac{\rho_p}{2}(e_p - E_p(\Theta))^2, \\ \qquad \text{if } e_p - E_p(\Theta) \leq \frac{\lambda_p}{\rho_p} \\ -\frac{(\lambda_p)^2}{2\rho_p}, \text{ otherwise.} \end{cases}$$

The superscript $k$ in $L_A^k(.)$ emphasizes the fact that the primary objective function that is being minimized is $e_k$ (and there are a total of $2 \cdot M$ of such augmented Lagrangian functions, $L_A^k$) with all other objectives being constrained.

The numerical minimization of $L_A^k(\Lambda, \lambda, \rho)$ at the $t^{th}$ iteration requires finding a direction of descent, $\mathbf{d}^{(t)}$, and an appropriate step-size, $\alpha^{(t)}$, in this direction. The quasi-Newton direction, $\mathbf{d}^{(t)} = -H^{(t)} \nabla_\omega L_A^k(\Lambda)$ is used here. The matrix $H^{(t)}$ is an approximation to the inverse of the Hessian matrix, $\nabla^2_{\omega\omega} L_A^k$, that is computed by the BFGS method. The step-size is searched by the Armijo rule, which suggests to start with a large step-size in the direction of $\mathbf{d}^{(t)}$ and try a smaller step-size as long as the objective is not reduced. Once the direction, $\mathbf{d}^{(t)}$, and an appropriate step-size, $\alpha^{(t)}$, are computed, the classifier parameters are updated as

$$\mathbf{w}_m^{(t+1)} = \mathbf{w}_m^{(t)} + \alpha^{(t)} \mathbf{d}^{(t)}, m = 1, 2, ..., M. \tag{7.2.4}$$

The augmented Lagrangian method requires that any deviation from the constraint bounds are prevented by the penalty parameter. In cases where the directional step-size yields an unacceptable deviation, the penalty parameter, $\rho$, is increased (say, by 10 times) and another step-size is searched for (as devised by Powell's method in Section 3.1.2) to ensure global convergence.

#### *7.2.2.2 Finding and Validating the Step-Size*

Let the current objectives be $f_1^{(t)}$ and $f_2^{(t)}$. Without loss of generality, assume that we are solving the subproblem where $f_1$ is being minimized and $f_2$ has a constraint on it. We allow $f_2$ to slightly increase from its current value to a value $f_2^{(t+1)} = f_2^{(t)} + \delta_2, \delta_2 > 0$ with the intention that $f_1$ will reduce substantially from its current value to a value $f_1^{(t+1)} = f_1^{(t)} - \delta_1, \delta_1 > 0$. All we want is that the increase in $f_2$ is compensated by the reduction in $f_1$, i.e., $\delta_1 > \delta_2$, in fact we want to have $\delta_1 \gg \delta_2$. In either case, this means that the sum $(f_1^{(t)} + f_2^{(t)})$ is greater than the sum $(f_1^{(t+1)} + f_2^{(t+1)})$ since

$$\begin{aligned} f_1^{(t+1)} + f_2^{(t+1)} &= f_1^{(t)} - \delta_1 + f_2^{(t)} + \delta_2 \\ &= [f_1^{(t)} + f_2^{(t)}] - [\delta_1 - \delta_2], \\ & \quad \delta_1 - \delta_2 > 0 \end{aligned} \tag{7.2.5}$$

Hence, the next iteration provides better objective values if the sum $(f_1 + f_2)$ reduces to a smaller value. This fact is our motivation to use *average error rate* as the utility function to judge the preferability of the new step found with the Armijo rule. That is, with the ICO method for the LID application, the step-size found by Armijo rule is validated and the classifier models are updated *only* in those cases which do not result in degradation in terms of average error, $E^{avg}(\Theta)$.

#### *7.2.2.3 Algorithm Description*

In this section, we present an algorithmic implementation of the proposed ICO framework. There are two main algorithms, namely the ICO-$K$ and MinimizeOneObjective algorithms. These two algorithms are summarized in Tables 10 and 11.

The ICO-$K$ algorithm minimizes $f_k$ subject to constraints on other objective functions, i.e., $f_p \leq \overline{f_p},\ p \neq k, p = 1, ..., K$. This in turn requires updating the parameters of all $M$ classifiers. This is achieved by the MinimizeOneObjective algorithm in each iteration of which the goal is to first find the quasi-Newton direction, $\mathbf{d}^{(t)}$. Upon finding $\mathbf{d}^{(t)}$, an appropriate step-size $\alpha^{(t)}$ in this direction is searched for by Armijo rule. The step-size should satisfy two conditions. First of all, any deviation from the constraints should be negligibly small. Otherwise, the penalty parameters, $\rho_p$'s, are increased and another step-size is searched for. Secondly, the step-size is validated only if $E^{avg}(\Theta)$ reduces or stays the same when the step is taken. Once a step-size that satisfies both of these conditions is found, the classifier parameters are updated. This is followed by the update of the approximation of inverse Hessian, $H^{(t)}$, and the Lagrange parameters, $\lambda_p$'s.

## *7.3 Experimental Results*

In this section, we report our experimental results on the LID task. Given a speech segment of duration 30 seconds and a target language, the task is to determine whether or not the speech is from the target language. The system designer needs to build 15 classifiers, which collectively classify unseen samples to the 12 main languages (no dialect identification is required). The target languages are ARABIC, ENGLISH, FARSI, FRENCH, GERMAN,

**Table 10:** ICO-K: ICO algorithm for $K$-language LID

I. **Start with:**

1. A set of $M$ LDFs with parameter vectors $\mathbf{w}_m \in \Re^n$.
2. A set of training set samples and development set samples.
3. A set of $K$ differentiable conflicting objectives $f_k$.
4. Sigmoid function parameters $\alpha > 0, \beta_m \in \Re$, $\eta > 0$.
5. Lagrange and penalty parameters $\lambda_m > 0$. $\rho_m \geq 0$

II. **Repeat (e.g., for 10 iterations):**

1. Classify development data to get $\mathbf{f}^{dev} = [f_1^{dev}, ..., f_K^{dev}]$.
2. Rank $f^{dev}$ in decreasing order so that $f_{(1)}^{dev} \geq f_{(2)}^{dev} \geq ... \geq f_{(K)}^{dev}$.
3. Classify training data to get $\bar{\mathbf{f}} = \{\bar{f}_1, ..., \bar{f_K}\}$.
4. We want to solve for each $k = 1, ..., K$:

$$\min f_{(k)}(\Theta)$$

$$\text{subject to } f_{(p)}(\Theta) \leq \overline{f}_{(p)}, p \neq k.$$

The objective function that has the worst performance on the development data comes the first. Hence, for each $k = 1, .., K$:

a. Choose $f_{(k)}$ to be minimized.

b. **Repeat for each** $p = 1, ..., K$.

i. Call MinimizeOneObjective algorithm to update $\mathbf{w}_{(p)}$.

ii. Classify the data to update $\bar{\mathbf{f}}$ vector.

**Table 11:** MinimizeOneObjective algorithm for LID

I. **Set** $t = 0$.

II. **Repeat until convergence:**

1. Calculate the gradient $\nabla_{\mathbf{w}} L_a^k$.
2. Calculate the quasi-Newton direction, $\mathbf{d}^{(t)}$.
3. We want to find a step-size $\alpha^{(t)}$ in the direction of $\mathbf{d}^{(t)}$. If such a direction is found, it means that we can improve $L_a^k$.

   However, we need to *validate* the found step-size:

   a. Check if the deviations from the constraints (if any) are acceptable. This can be achieved by calculating the norm of the slack vector $\|c\|$. As Powell's method devises, if $\|c\| > \frac{1}{4}\|c\|$, set $\rho = 10\rho_t$ and go to Step II.1.

   b. Check if the new step results in a more desirable indifference curve. This can be evaluated by comparing the local utilities, $u$.

      i. Set $u^0 = u$.

      ii. Update $H^{(t)}$ by the BFGS update rule.

      iii. Update $\lambda$ by Powell's method.

      iv. Save the updated $\mathbf{w}_{(p)}$.

Hindi, Japanese, Korean, Mandarin, Spanish, Tamil, and Vietnamese. Three of the 12 languages (English, Mandarin, and Spanish) contain material for two dialects.

### 7.3.1 LID Data

The training data set was organized for a total of 15 languages and dialects. Each language or dialect had roughly 11,000 training samples. The total size of the training set was 177,229 samples. The 1996 test set, which we used as the test set to evaluate the performance of the systems, has a total of 1492 samples, where each language has roughly 80 samples except for English, which has 478 samples.

### 7.3.2 Score Distribution Feature Vector

Although the extraction of feature vectors is beyond the scope of this thesis, we give a brief description. Instead of using frame-based vectors (such as MFCCs, PLP, or SDC features) as the front end features in most conventional LID systems, Bin et al. [61] extracted utterance-based score vectors generated by parallel phone recognizers followed by language models (P-PRLM) [113] and bag of sounds (BOS) [66] models. For constructing feature vectors, 39-dimensional MFCCs were extracted from each frame. Utterance based cepstral mean subtraction was applied to the MFCC features to remove channel distortion. Each phoneme in the languages was modeled with a 3-state HMM.

Seven phone recognizers were built: English, Korean, Mandarin, Japanese, Hindi, Spanish, and German. English phonemes are trained from IIR-LID corpus [1]. Korean phonemes are trained from LDC Korean corpus (LDC2003S03). Mandarin phonemes are trained from the MAT corpus [106]. Other phonemes are trained from OGI-TS corpus [2].

The 15-language/dialect training database is tokenized into a collection of text-like phone sequences from each of the 7 tokenizers. The P-PRLM scores were computed based on the resulting phone sequences. Then, a trigram phone LM was trained for each P-PRLM tokenizer-target language pair.

The BOS method uses a universal sound recognizer to tokenize an utterance into a phone

[1] Available at http://sdp.i2r.a-star.edu.sg.
[2] Available at http://cslu.cse.ogi.edu/corpora/corpCurrent.html.

sequence, which was then converted into a count vector [66]. The universal sound inventory was a combined phoneme set from six languages: ENGLISH, MANDARIN, JAPANESE, HINDI, SPANISH, and GERMAN. There were 258 phonemes in total. For each phone sequence generated from the universal sound tokenizer, we counted the occurrence of bi-phones. A phone sequence was then represented as a vector of bi-phone occurrence with $258 \times 258 = 66,564$ elements. We trained pair-wise SVM classifiers for all of the target languages using a linear kernel. Finally, the PPRLM and the BOS scores were concatenated to form the feature vectors, $x$.

### 7.3.3 Comparison of SOP- and MOP-Based Approaches

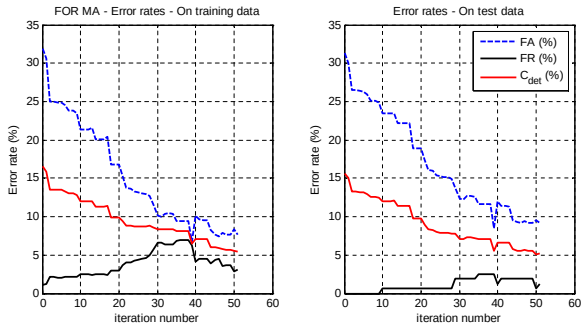
In our experiments, $\eta$ (see (7.1.1)) was 7, $\alpha$ (see (7.1.1)) was 1, and $\beta_m$ (see (7.1.1)) was heuristically adjusted for each class. The heuristics was such that the $\pi_m(x, \Lambda)$ that are small in magnitude are summed and averaged. The resulting average was used as $\beta_m$. The idea behind the adopted heuristics is to associate more loss to those samples for which $\ell_m(d_m(x, \Lambda))$ is close to 0.5. Such samples represent the more confusable examples to the classifier. In our experiments, for generating constraint bounds as in (5.1.2), we set $\delta = (10^{-3}, ..., 10^{-3})$.

#### *7.3.3.1 ICO with Two Objectives*

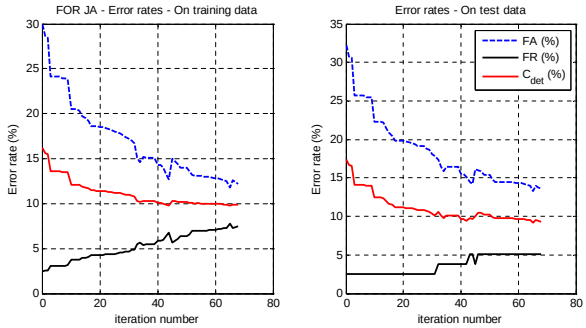
One approach to solving the LID problem is to design independent binary classifiers to achieve a tradeoff between the two conflicting metrics for each language, $E^{FR}(\Theta)$ and $E^{FA}(\Theta)$.

In Figure 19(a), the changes of $E^{FR}$, $E^{FA}$, and $E^{avg}$ for both the training and test data are illustrated for the language MANDARIN. An inspection of the change of errors rates shows that $E^{FA}$ was reduced from 31.90% to 7.69% (i.e., by more than 24%) at the expense of an increase in the $E^{FR}$ from 1.11% to 3.09% (i.e., by less than 2%) in the training data. Similarly, in the test data, $E^{FA}$ was reduced from 31.29% to 9.21% (i.e., by more than 22%) at the expense of an increase in $E^{FR}$ from 0.00% to 1.28%. Eventually, the two conflicting objectives reached a balance. Similar results were obtained for the other languages, and further experiments, as illustrated in Figures 19(b) and 19(c), support our argument that

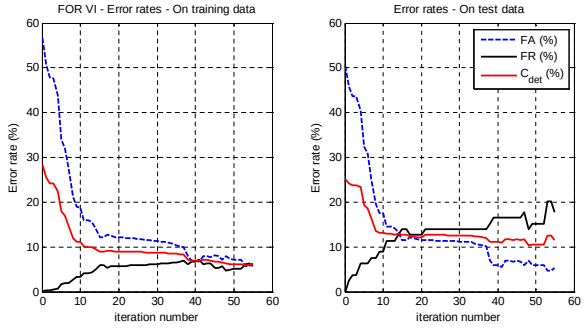
the improvement in one objective is obtained only with a slight degradation in others.




(a) For the language MANDARIN




(b) For the language JAPANESE




(c) For the language VIETNAMESE

**Figure 19:** The individual error rates move toward a better balance at each iteration.

In Figure 20, we plotted DET curves for the language MANDARIN when two detection systems, one SOP-trained and one ICO-trained, were used. Furthermore, in Table 12, we report the point in the DET curve where the average of FR and FA rates was minimized. Several conclusions in favor of ICO-trained classifiers are drawn from the DET curves in Figure 20 and the results in Table 12:

- Although the SOP classifiers were trained so that an *equally-weighted* sum of FR and FR rates was minimized, the individual error rates at the point where their average

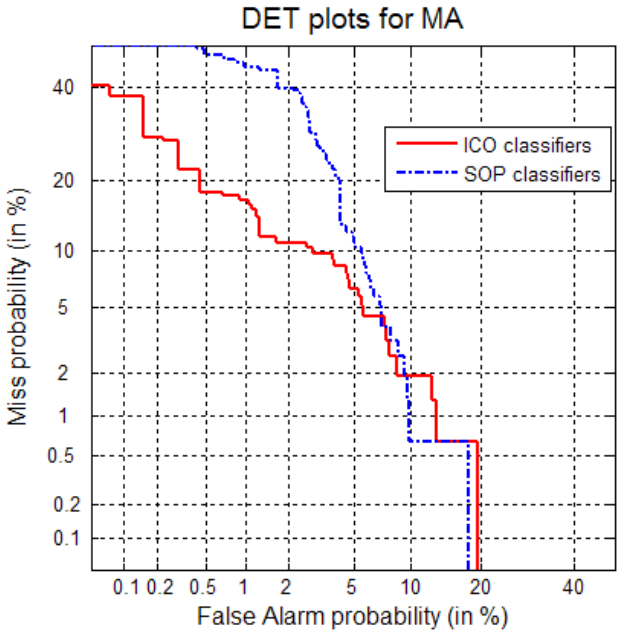



**Figure 20:** The DET curves for MANDARIN.

**Table 12:** The FR and FA rates on the DET curve when the average error rate is at its minimum.

| | FR (%) | FA (%) | Average (%) |
|---|---|---|---|
| SOP classifiers | 0.64 | 9.83 | 5.24 |
| ICO classifiers | 4.49 | 5.69 | 5.09 |

was minimal were highly unbalanced: The FA rate was 9.83%, while the FR rate was only 0.64%. Meantime, minimizing the average error rate over the training samples did not generalize well to the test samples.

- At the point of minimal average error rate, the ICO-trained classifiers yielded an FA rate of 5.69%, and an FR rate of 4.49%. Hence, as was our motivation in this research, ICO training was successful at achieving a good balance between the two conflicting objectives.

- As can be observed from Figure 20, the equal error rate of the ICO-trained classifiers is also lower than that of the SOP-trained classifiers.

- From Figure 20, we observe that when the FR rate is lower than 2%, which corresponds to classifying 3 MA samples as non-MA samples, SOP-trained classifiers performed slightly better. For all other FR rates, ICO-trained classifiers performed better. Besides, the DET curve of SOP-trained classifiers were sharper than that of ICO-trained classifiers, which suggests that when we attempt to only slightly reduce the FA rate, the FR rate degrades dramatically.

- When the system design requires a tradeoff among error types, a single performance number is inadequate to represent the capabilities of a system. This is because such a system has many operating points, and is best represented by a performance curve. Therefore, optimization of Receiver Operating Characteristics (ROC), and in particular the maximization of area under the ROC curve (AUC), has taken considerable research interest [63, 111]. Maximization of AUC corresponds to minimizing the area under the DET curves in Figure 20. Judging from Figure 20, it is concluded that the ICO-trained classifiers perform better in the overall and more robust than the SOP-trained classifiers.

#### *7.3.3.2 ICO with More Than Two Objectives*

In Table 13, the individual error rates for the SOP-trained and the ICO-trained classifiers for each language are reported. There are different ways to look at the results in Table 13.

*Analyzing the Summary Statistics*

As a first remark, both the average FA and FR rates are significantly reduced by the ICO training with 17.6% and 16% relative improvement, respectively. Secondly, the outlier objective values are *smoothed*. This is evidenced by the upper and lower 10% percentile averages are reported in the bottom two rows of Table 13. The lower 10% percentile average is lower for the SOP- than for the ICO-trained classifiers, and the upper 10% percentile average is higher for SOP- than for the ICO-trained classifiers. This verifies that there are some objective functions which are favored or sacrificed by the SOP training.

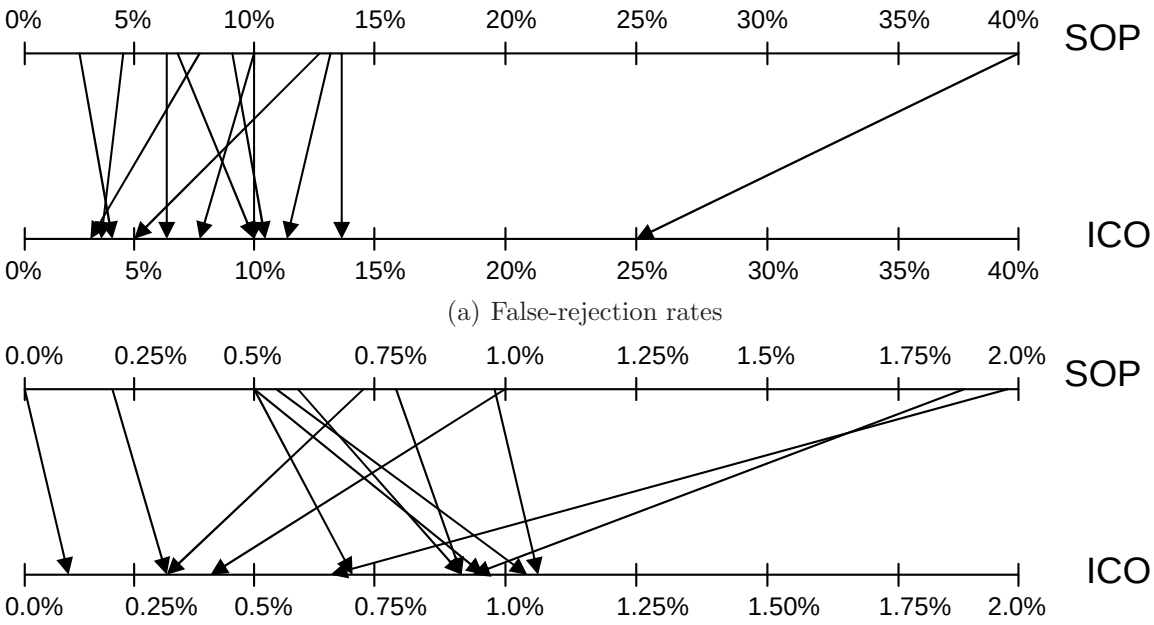
For clarity, we have also mapped the error rates in Table 13 into a portion of the real

**Table 13:** Compromise solutions for the SOP- and ICO-trained classifiers

| | SOP Classifiers | | ICO Classifiers | |
|---|---|---|---|---|
| Language | FR(%) | FA(%) | FR(%) | FA(%) |
| Arabic | 12.50 | 0.21 | 5.00 | 0.28 |
| English | 2.72 | 1.97 | 3.97 | 0.69 |
| Farsi | 10.00 | 0.64 | 7.50 | 0.85 |
| French | 7.50 | 0.50 | 3.75 | 0.92 |
| German | 6.25 | 0.78 | 6.25 | 0.85 |
| Hindi | 39.47 | 0.57 | 25.00 | 1.13 |
| Japanese | 6.33 | 0.71 | 10.12 | 0.28 |
| Korean | 10.26 | 1.84 | 10.26 | 0.92 |
| Mandarin | 4.49 | 0.97 | 3.85 | 1.20 |
| Spanish | 9.15 | 0 | 10.46 | 0.07 |
| Tamil | 13.70 | 0.49 | 12.33 | 0.7 |
| Vietnamese | 13.92 | 0.99 | 13.92 | 0.42 |
| Average | 11.36 | 0.81 | 9.37 | 0.69 |
| Upper 10% Percentile Avg. | 26.70% | | 19.46% | |
| Lower 10% Percentile Avg. | 0.11% | | 0.17% | |

line as in Figure 21. We have not plotted the FR and FA rates on the same graph since their scales are very different. Nevertheless, it should be kept in mind that each of the FR and FA rates compete against all others. It is depicted in Figure 21(b) that the major outlier FR rate was recorded for Hindi with a value of 39.47%. The ICO algorithm was able to drop it to 25% with only (relatively) small changes in the other objectives. Since any improvement in one objective comes at the expense of degradation in some other FR or FA rates, some FR and FA rates increased as expected.

As shown in Figure 21(b), the FR rates for the SOP classifiers range from 2.72% to 39.47%. With ICO training, this wide range was reduced to a range from 3.75% to 25.00%. Hence, apparently, the best FR rates are increased while the worst ones are *greatly* reduced, i.e., outliers are smoothed. This experimental finding supports our argument that classifiers with a single training objective that is an accumulation of several conflicting objectives are not good at resulting in acceptable objective values for individual ones. In contrast, ICO targets at achieving a better balance among the competing objectives. Similar arguments are valid for FA rates as well, as shown in Figure 21(b).




(a) False-rejection rates




(b) False-acceptance rates

**Figure 21:** Those error rates that are the smallest or the largest (i.e., outliers) are smoothed so that the differences between different error rates are less dramatic.

*Analyzing the Individual Error Rates*

The improvements in the individual error rates when ICO, instead of SOP, was used as learning goal are noteworthy. For example, the FR rate for ARABIC was reduced from 12.50% to 5%, and the FR rate for HINDI was reduced from 39.47% to 25%. The relative improvements for these two languages are 60% and 36.7%, respectively. On the other hand, we witness an increase in the FR rates for ENGLISH, JAPANESE, and SPANISH. As already emphasized, the improvements in some objectives come at the expense of some degradation in some others. It is also noted in the last column of Table 13 that whenever an FR rate for a language is increased relative to the first column, its corresponding FA rate reduces and vice versa, as our analysis suggests.

# CHAPTER VIII

# CONCLUSIONS AND FUTURE WORK

Most engineering problems involve the consideration of a tradeoff among many conflicting design objectives. The common approach is to set a single overall objective function that can combine all the objectives and to optimize it through the use of classical single-objective programming (SOP) methods. However, there is no guarantee on the performance of the individual objectives since in most cases the overall objective function overlooks the underlying tradeoffs between the conflicting objectives.

The problems that involve complex tradeoff relationships among many conflicting objectives are the subject of multi-objective programming (MOP). In this dissertation we investigated MOP-based methods for speech and natural language applications, including statistical language model (LM) adaptation and automatic language identification (LID). We presented a formulation of an MOP-based approach, called iterative constrained optimization (ICO). The original optimization problem with conflicting objectives is formulated as an iterative process of single-objective problems in which the remaining objectives are embedded as constraints. The bounds needed to constrain the conflicting objectives were adjusted in an iterative manner. For each of the two applications, we first considered an SOP-based approach. We, then, formulated the original problem as an MOP problem and solved it using the ICO method. Finally, we compared the performance of the SOP- and MOP-based solutions for each of the applications.

## *8.1 Summary of Major Contributions*

This dissertation makes contributions to the problem of learning parameters of a statistical model from the following three aspects:

- This dissertation investigates the use of MOP in speech and language applications, which is not well-explored. MOP is in fact well-suited for similar applications because

  - incommensurate objectives arising in similar applications can easily be handled,
  - the range of obtainable objective functions is wider and no opportunity is missed,
  - the solution methodologies are intuitive.

- This dissertation is expected to provide new insights and techniques for accomplishing significant performance improvement over existing approaches in terms of the individual competing objectives. Meantime, the designer has a better control over what is achieved in terms of the individual objectives.

- Most of the research attempts that aim at using MOP in similar applications illustrate their methods in problems with two or three objectives. This dissertation reports experimental results for as many as 30 conflicting objectives in automatic language identification and four objectives in statistical language model adaptation.

## *8.2 Conclusions drawn from Statistical LM Adaptation*

There is an intrinsic detail versus reliability tradeoff in every statistical modeling problem. Language model (LM) adaptation is a problem in which the resulting LM is formed so that it is close to both a model derived from a general domain corpus (for achieving reliability) and another model derived from limited application-specific data (for having sufficient detail).

In Chapter 6, we first developed an SOP-based reference model, called structural MAP (SMAP). We then reformulated the training objective of SMAP so that the conflicting nature of the problem was handled by two separate objectives. Doing so resulted in a target LM in the form of a log-linear interpolation of component LM probabilities.

Several important conclusions were drawn from our experimental investigations including:

- The SMAP hierarchical model for LM training and adaptation made it possible to estimate the prior and the posterior probabilities of the $n$-grams in a top-down manner. This was useful since the child nodes can make effective use of information coming from parent node in the face of data sparsity. Inheriting information from parent

nodes provides a mechanism to remedy the data sparseness problem especially when high order $n$-grams are used.

- The hierarchical structure in SMAP made it possible to merge the information coming from the general domain corpus and the information coming from the application-specific data on individual nodes.

- SMAP can easily be extended to higher order $n$-grams. Moreover, it is possible to grow very deep trees and then to prune different trees to different $n$ depending on the available data for the corresponding root node (i.e., unigram). Hence, we can easily form variable-length $n$-gram models.

- The average KL divergences of the target model from two independently models, one trained on the general domain data and the other trained on the application-specific data, were being minimized when ICO was used for LM adaptation. In experiments, the average KL divergences were shown to move towards a better balance. Moreover, each KL divergence followed a zigzag pattern throughout the iterative process. This is because the two objectives attempt to modify the target model probabilities to be close to their respective model.

- In our ASR experiments with ICO for LM adaptation, the perplexities and the WERs were only slightly different from reference models, but the best results were obtained with the ICO LM adaptation method. The major gains in solving the LM adaptation problem with MOP are twofold:

  - Upon observing the behavior of each objective, we have full freedom to tune the system into different operating points to meet different requirements.

  - Meantime, we can easily avoid extremes, i.e., the cases that the target LM is too dependent on the application specific data or on the general domain data.

- It was found that the MOP-based approach is superior to SMAP in terms of WER in ASR experiments. This is because MOP leaves more flexibility in finding the $n$-gram

estimates while SMAP attempts to merge the conflicting goals a priori in an overall function.

- Most importantly, the proposed approach to LM adaptation stresses the richness of MOP for use in a variety of problems that we encounter in natural language processing. A different MOP-based formulation of LM adaptation could yield another useful technique and this line of research can be furthered into other MOP-based techniques.

## *8.3 Conclusions drawn from LID*

In the LID application, we would like to identify each language correctly, which requires the simultaneous minimization of false-acceptance (FA) and false-rejection (FR) rates of each of the target languages. Using conventional SOP techniques, $E^{avg}$ was directly minimized as a baseline system. Following that, each FA rate and each FR rate for different languages was taken as a separate objective and ICO was used to iteratively optimize each error rate. Several conclusions were drawn from this application:

- When independent binary classifiers were trained for each language, the individual FA and FR rates were highly unbalanced at the point where their average was minimal.

- Both the average FA and FR rates are significantly reduced by the ICO training respectively.

- One of the most important conclusions was that the outlier objective values were smoothed. There were some objective functions which were favored or sacrificed by the SOP training. This experimental finding supports our argument that classifiers with a single overall training objective are not good at resulting in acceptable objective values for individual ones. In contrast, ICO targets at achieving a better balance among the competing objectives.

- The DET curve of SOP-trained classifiers were sharper than that of ICO trained classifiers, which suggests that when we attempt to only slightly reduce the FA rate, the FR rate degrades dramatically.

- Consequently, we illustrated the use of ICO on automatic language identification where the best compromise solution among a total of 30 competing objectives was searched for. Our experimental results indicated that SOP-trained classifiers are not good at resulting acceptable objective values for individual ones. In contrast, ICO training is capable of achieving a better balance among the competing objectives.

## *8.4 Future Research Directions*

In our future work, we will extend our experimental results to 4-gram and 5-gram LMs. Inheriting information from parent nodes provides a mechanism to remedy the data sparseness problem especially when high order $n$-grams are used. It is also possible to make reasonable predictions for histories that we have not previously seen by using the class-based language modeling paradigm. The SMAP LM adaptation can also be used in applications other than ASR. For instance, it can be incorporated into a spoken utterance classification system to find $n$-gram probabilities that minimize the WER and/or CER.

In Chapter 6, ICO was used for LM adaptation to combine information from two different domains. The same could be done to merge information from a larger number of domains. Thus, ICO could be used to perform topic-based LM adaptation. In addition to ICO, a different MOP-based formulation of LM adaptation could yield another useful technique and this line of research can be furthered into other MOP-based techniques.

We believe ICO is well suited for many problems in a wide range of applications. This line of research can be furthered in several theoretically rich directions. One of the first ones is about the automatic means to infer the constraint bounds instead of using 'magical' numbers. We have observed in our experiments that different settings for the constraint bounds directly translate into different end results, as intuition suggests. Based on our experience, we foresee that the constraint bounds can be set by analyzing the sensitivity of the problem on the changes of the individual objectives.

Many problems, including automatic summarization, and spoken language understanding, can be posed as MOP problems. MOP-based approaches are especially well-suited for problems in which it is not obvious how to combine the incommensurate objectives

into an overall objective function. For instance, building effective summaries requires the minimization of summary length, the maximization of inclusion of different features, and the maximization of the information content per feature. All such incommensurate objective functions can be efficiently optimized with ICO to achieve satisfactory levels for each objective.

# REFERENCES


[1] Andersson, J., "A survey of multiobjective optimization in engineering design," Technical report LiTH-IKP-R-1097, Department of Mechanical Engineering, Linkping University, Linkping, Sweden, 2000.

[2] Anthony, M. and Bartlett, P. L., *Neural Network Learning: Theoretical Foundation*. Cambridge University Press, 1999.

[3] Bacchiani, M., Roark, B., and Saraclar, M., "Language Model Adaptation with MAP Estimation and the Perceptron Algorithm," in *the Proc. of the Human Language Technology Conference*, (Boston, MA), May 2004.

[4] Bahl, L. R., Brown, P. F., Souza, P. V., and Mercer, R. L., "A tree-based statistical language model for natural language speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 7, pp. 1001–1008, 1989.

[5] Bellegarda, J. R., "A Multi-Span Language Modeling Framework for Large Vocabulary Speech Recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 5, pp. 456–467, 1998.

[6] Berger, J. O., *Statistical Decision Theory and Bayesian Analysis*. Springer, 1993.

[7] Bertsekas, D. P., *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[8] Besling, S. and Meier, H. G., "Language model speaker adaptation," in *the Proc. of InterSpeech - European Conference on Speech Communication and Technology*, (Madrid, Spain), pp. 1755–1758, 1995.

[9] Bielefeld, B., "Language identification using shifted delta cepstrum," in *Fourteenth Annual Speech Research Symposium*, 1994.

[10] Bishop, C. M., *Pattern Recognition and Machine Learning*. Springer, 2006.

[11] Braga, A. P., Takahashi, R. H. C., Costa, M. A., and Teixeira, R. A., *Multi-Objective Machine Learning*, ch. Multi-Objective Algorithms for Neural Networks Learning, pp. 151–171. Berlin, Heidelberg: Springer, 2006.

[12] Brown, P. F., Pietra, V. J. D., deSouza, P. V., Lai, J. C., and Mercer, R. L., "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[13] Charnes, A., Cooper, W. W., and Ferguson, R. O., "Optimal estimation of executive compensation by linear modeling," *Management Science 1:2*, vol. 1, pp. 138–151, 1955.

[14] Chase, L., Rosenfeld, R., and Ward, W., "Error-responsive modifications to speech-recognisers: negative n-grams," in *the Proc. of the International Conference on Spoken Language Processing*, (Yokohama), pp. 827–830, 1994.

[15] Chelba, C., "Portability of syntactic structure for language modeling," in *the Proc. of International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, (Salt Lake City, UT), pp. a–d, May 2001.

[16] Chien, J.-T. and Wu, M.-S., "Adaptive Bayesian latent semantic analysis," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 1, pp. 198–207, 2008.

[17] Chou, W., Lee, C.-H., and Juang, B.-H., "Segmental GPD Training of an Hidden Markov Model based Speech Recognizer," in *the Proc. of International Conference on Acoustics, Speech and Signal Processing*, (San Francisco, CA), 1992.

[18] Cimarusti, D. and Ives, R. B., "Development of an automatic identification system of spoken languages: Phase 1," in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, (Paris, France), May 1982.

[19] Coello, C. A. C., "A Short Tutorial on Evolutionary Multiobjective Optimization," in *First International Conference on Evolutionary Multi-Criterion Optimization* (Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C., and Corne, D., eds.), pp. 21–40, Springer-Verlag. Lecture Notes in Computer Science, 2001.

[20] Cover, T. M. and Thomas, J. A., *Elements of Information Theory.* John Wiley & Sons, 1991.

[21] Da Cunha, N. O. and Polak, E., "Constrained minimization under vector-valued criteria in finite dimensional spaces," *Journal of Mathematical Analysis and Applications*, vol. 19, no. 1, pp. 103–124, 1967.

[22] Darroch, J. N. and Ratcliff, D., "Generalized Iterative Scaling for Log-Linear Models," *The Annals of Mathematical Statistics*, vol. 43, no. 5, pp. 1470–1480, 2000.

[23] Dempster, A. P., Laird, N. M., and Rubin, D. B., "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 38, no. 1, pp. 1–38, 1977.

[24] Duda, R., Hart, P., and Stork, D., *Pattern Classification.* John Wiley & Sons, 2001.

[25] Federico, M., "Bayesian estimation methods for n-gram language model adaptation," in *the Proc. of International Conference on Spoken Language Processing*, (Philadelphia, PA), p. 240 243, October 1996.

[26] Fishburn, P. C., *Utility theory for decision making.* New York, Wiley, 1970.

[27] Fletcher, R., *Practical Methods of Optimization.* John Wiley & Sons, 2000.

[28] Foil, J. T., "Language identification using noisy speech," in *the Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, (Tokyo, Japan), 1986.

[29] Freitas, A. A., "A Critical Review of Multi Objective Optimization in Data Mining," *ACM SIGKDD Explorations Newsletter*, vol. 6, pp. 1345–1348, December 2004.

[30] Gales, M. and Woodland, P., "Recent advances in large vocabulary continuous speech recognition: An htk perspective," in *Tutorial in International Conference on Acoustics, Speech, and Signal Processing*, (Toulouse, France), 2006.

[31] Gauvain, J.-L. and Lee, C.-H., "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[32] Glover, F., "Tabu search - part i," *ORSA Journal on Computing*, vol. 1, pp. 190–206, 1989.

[33] Good, I. J., "The population frequencies of species and the estimation of population parameters," *Biometrika*, vol. 40, no. 3 and 4, pp. 237–264, 1953.

[34] Handl, J., Kell, D. B., and Knowles, J., "Multiobjective Optimization in Bioinformatics and Computational Biology," *IEEE/ACM Transactions On Computational Biology and Bioinformatics*, vol. 4, no. 2, pp. 279–293, 2007.

[35] Hazen, T. and Zue, V. W., "Automatic language identification using a segment-based approach," in *the Proc. of European Conference on Speech Communication and Technology*, 1993.

[36] Holland, H. J., *Adaptation in Natural and Artificial Systems, an introductory analysis with application to biology, control and artificial intelligence.* Ann Arbor, MI: The university of Michigan Press, 1975.

[37] Huo, Q. and Lee, C.-H., "On-line adaptive learning of the continuous density hidden Markov model based on approximate recursive Bayes estimate," *IEEE Transactions on Speech and Audio Processing*, vol. 5, pp. 161–172, March 1997.

[38] Hutchins, S. E. and Thyme-Gobbel, A., "Experiments using prosody for language identification," in *the Proc. of Speech Research Symposium*, (Baltimore, MD), 1994.

[39] Itahashi, S. and Du, L., "Language identification based on speech fundamental frequency," in *the Proc. of the Fourth European Conference on Speech Communication and Technology*, vol. 2, (Madrid, Spain), pp. 1359–1362, September 1995.

[40] Iyer, R. and Ostendorf, M., "Modeling long distance dependencies in language: Topic mixtures versus dynamic cache models," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 51–62, 1999.

[41] Jaynes, E. T., "Prior probabilities," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, pp. 227–241, September 1968.

[42] Jelinek, F., "The development of an experimental discrete dictation recognizer," *the Proceedings of IEEE*, vol. 73, no. 11, pp. 1616–1624, 1985.

[43] Jelinek, F. and Mercer, R. L., "Up from trigrams: the struggle for improved language models," in *the Proc. of Eurospeech*, (Genoa), pp. 1037–1040, 1991.

[44] Jensen, M. T., "Guiding single-objective optimization using multi-objective methods," *Applications of Evolutionary Computation*, pp. 268–279, 2003.

[45] Johnson, N. L. and Kotz, S., *Distributions in Statistics*. New York: Wiley, 1972.

[46] Johnson, W. E., "Probability: deductive and inductive problems," *Mind*, vol. 41, pp. 421–423, 1932.

[47] Juang, B.-H. and Katagiri, S., "Discriminative learning for minimum error classification," *IEEE Transactions on Speech and Audio Processing*, vol. 40, p. 30433054, December 1992.

[48] Katz, S. M., "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, pp. 400–401, March 1987.

[49] Keeney, R. L. and Raiffa, H., *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge, UK: Cambridge University Press, 1993.

[50] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[51] Klakow, D., "Log-linear interpolation of language models," vol. 5, (Sydney, Australia), pp. 1695–1699, 1998.

[52] Knowles, J. D., Watson, R. A., and Corne, D. W., "Reducing local optima in singleobjective problems by multi-objectivization," in *the Proc. of International Conference on Evolutionary Multi-Criterion Optimization*, (Berlin, Germany), 2001.

[53] Kuhn, H. W. and Tucker, A., "Nonlinear programming," in *2nd Berkeley Symp. on Mathematical Statistics and Probability* (Neyman, J., ed.), University of California Press, Berkeley, 1951.

[54] Kuhn, R. and de Mori, R., "A cache-based natural language method for speech recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 6, pp. 570–582, 1990.

[55] Kuhn, R. and de Mori, R., "A cache-based natural language model for speech recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 570–583, June 1990.

[56] Kullback, S., *Information theory and statistics*. John Wiley & Sons, 1959.

[57] Kullback, S. and Leibler, R. A., "On Information and Sufficency," *The Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951.

[58] Lander, T., Cole, R. A., Oshika, B. T., and Noel, M., "The OGI 22 language telephone speech corpus," in *the Proc. of the Fourth European Conference on Speech Communication and Technology*, (Madrid, Spain), September 1995.

[59] (LDC), L. D. C., "Callhome and callfriend lvcsr databases," 1996.

[60] Li, H., Ma, B., and Lee, C.-H., "A vector space modeling approach to spoken language identification," *IEEE Transantions on Audio, Speech, and Language Processing*, vol. 15, pp. 271 – 284, January 2007.

[61] Li, J., Yaman, S., Lee, C.-H., Ma, B., Tong, R., Zhu, D., and Li, H., "Language recognition based on score distribution feature vectors and discriminative training," in *the Proc. of IEEE Odyssey: The Speaker and Language Recognition Workshop*, (San Juan, Puerto Rico), 2006.

[62] Lidstone, G. J., "Note on the general case of the bayes-laplace formula for inductive or a posteriori probabilities," *Transactions of the Faculty of Actuaries*, vol. 8, pp. 182–192, 1920.

[63] Lim, S. G. C.-H. L. J. H., "An ensemble classifier learning approach to ROC optimization," in *the Proc. of International Conference on Pattern Recognition*, vol. 2, (Hong Kong, China), pp. 679 – 682, August 2006.

[64] Liu, G. P. and Kadirkamanathan, V., "Learning with Multi Objective Criteria," in *Proc. of IEE Conference on Artificial Neural Networks*, pp. 53–58, 1995.

[65] Lugosi, G. and Zeger, K., "Concept learning using complexity regularization," *IEEE Transactions on Information Theory*, vol. 42, no. 1, 1996.

[66] Ma, B., Li, H., and Lee, C.-H., "An acoustic segment modeling approach to automatic language identification," in *the Proc. of InterSpeech - European Conference on Speech Communication and Technology*, (Lisbon, Portugal), pp. 2837–2840, 2005.

[67] Martin, A., Doddington, G., Kamm, T., Ordowski, M., and Przybocki, M., "The det curve in assessment of detection task performance," in *the Proc. of Eurospeech*, (Rhodes, Greece), pp. 1895–1898, 2000.

[68] Marusciac, I., "On fritz john type optimality criterion in multi-objective optimization," *Reveu d'Analyse Numerique et de Theorie dl l'Approximation*, vol. 1, no. 1-2.

[69] Masataki, H., Sagisaka, Y., and Tawahara, T., "Task adaptation using MAP estimation in n-gram language model," in *the Proc. of International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 783–786, May 1997.

[70] Mattson, C. A., Mullur, A. A., and Messac, A., "Smart pareto filter: obtaining a minimal representation of multiobjective design space," *Engineering Optimization*, pp. 721–740, 2004.

[71] Miettinen, K., *Nonlinear Multiobjective Optimization*. Springer, 1999.

[72] Miettinen, K. and Mkel, M., "Interactive bundle-based method for nondifferentiable multiobjective optimization: Nimbus," *Optimization*, vol. 34, pp. 231–246, 1995.

[73] Minsky, M. and Papert, S. A., *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press, expanded edition - 1988/1969.

[74] Mitchell, T. M., *Machine Learning*. McGraw-Hill, 1997.

[75] Muthusamy, Y. K., *A segmental approach to Automatic Language Identification.* PhD thesis, Oregon Graduate Institute of Science and Technology, 1993.

[76] Muthusamy, Y. K., Barnard, E., and Cole, R. A., "Automatic language identification: A review-tutorial."

[77] Muthusamy, Y. K. and Cole, R. A., "Automatic segmentation and identification of ten languages using telephone speech," in *the Proc. of International Conference on Spoken Language Processing*, (Banf, Alberta, Canada), October 1992.

[78] Muthusamy, Y. K., Cole, R. A., and Oshika, B. T., "The OGI multi-language telephone speech corpus," in *the Proc. of International Conference of Spoken Language Processing*, (Alberta, Canada), 1992.

[79] Neyman, J. and Pearson, E., "On the problem of the most efficient tests of statistical hypotheses," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 231, pp. 289–337, 1933.

[80] Nocedal, J. and Wright, S. J., *Numerical Optimization.* Springer, 1999.

[81] Paul, D. B. and Baker, J. M., "The design for the Wall Street Journal based CSR corpus," in *the Proc. of Internation Conference of Spoken Language Processing*, (Banff, Alberta, Canada), pp. 899–902, September 1992.

[82] Paul, D. B. and Baker, J. M., "The design for the wall street journal-based CSR corpus," in *the Proc. of International Conference on Spoken Language Processing*, pp. 899–902, 1992.

[83] Pietra, S. D., Pietra, V. D., Mercer, R., and Roukus, S., "Adaptive language model estimation using minimum discriminant estimation," in *the Proc. of International Conference on Acoustics, Speech, and Signal Processing*, pp. 633–636, 1992.

[84] Rabiner, L. R., "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–285, February 1989.

[85] Rao, P. S., Dharanipragada, S., and Roukos, S., "MDI adaptation of language models across corpus," in *the Proc. of International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 161–164, 1995.

[86] Robinson, A., Hochberg, M., and Renals, S., "Ipa: Improved phone modelling with recurrent neural networks," in *the Proc. of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 37–40, 1994.

[87] Rosenfeld, R., *Adaptive Statistical Language Modeling: A Maximum Entropy Approach.* PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1994.

[88] Rosenfeld, R., "Optimizing lexical and n-gram coverage via judicious use of linguistic data," in *the Proc. of InterSpeech - European Conference on Speech Communication and Technology*, (Madrid, Spain), pp. 1763–1766, 1995.

[89] Rosenfeld, R. and Huang, X., "Improvements in stochastic language modeling," in *the Proc. of DARPA Speech and Language Workshop*, (San Mateo, CA), February 1992.

[90] Ruhlen, M., "A guide to the languages of the world." Stanford University, 1975.

[91] Saber, H. and Ravindran, A., "Nonlinear goal programming theory and practice: A survey," *Computers and Operations Research 20:3*, vol. 20:3, 275-271, 1993.

[92] Scholkopf, B. and Smola, A. J., *Learning with kernels*. The MIT Press, 2001.

[93] Seymore, K. and Rosenfeld, R., "Using story topics for language model adaptation," in *the Proc. of European Conference on Speech Communication and Technology*, vol. 4, (Rhodes, Greece), pp. 1987–1990, September 1997.

[94] Shannon, C. E., "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379423, 623656, 1948.

[95] Shannon, C. E. and Weaver, W., *The mathematical theory of communication*. Urbana: University of Illinois Press, 1949.

[96] Shaw, K. J., Nortcliffe, A. L., Thompson, M., Love, J., Fleming, P. J., and Fonseca, C. M., "Assessing the performance of multiobjective genetic algorithms for optimization of a batch process scheduling problem," in *the Proc. of IEEE Congress on Evolutionary Computation*, pp. 37–45, 1999.

[97] Shinoda, K. and Lee, C.-H., "A structural Bayes approach to speaker adaptation," *IEEE Transactions on Speech and Audio Processing*, vol. 9, pp. 276–287, March 2001.

[98] Siohan, O., Myrvoll, T. A., and Lee, C.-H., "Structural maximum a posteriori linear regression for fast HMM adaptation," *Computer Speech and Language*, vol. 16, pp. 5–24, January 2002.

[99] Suttorp, T. and Igel, C., *Multi-Objective Machine Learning*, ch. Multi-objective optimization of support vector machines, pp. 199–220. Berlin Heidelberg: Springer, 2006.

[100] Torres-Carrasquillo, P. A., Reynolds, D. A., and Deller, J. R., "Language identification using gaussian mixture model tokenization," in *the Proc. of International Conference of Acoustics, Speech, and Signal Processing*, (Orlando, FL), 2002.

[101] Torres-Carrasquillo, P. A., Singer, E., Reynolds, M. A. K. D. A., and Deller, J. R., "Approaches to language identification using gaussian mixture models and shifted delta cepstral features," in *the Proc. of International Conference on Spoken Language Processing*, (Denver, CO), pp. 33–36, September 2002.

[102] V., C. and Haimes, Y. Y., *Multiobjective Decision Making Theory and Methodology*. Elsevier Science Publishing Co., New York, 1983.

[103] Vaissiere, J., *Prosody: Models and Measurements*, ch. Language-independent prosodic features, pp. 53–66. Springer-Verlag, 1983.

[104] Vapnik, V., *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[105] Vapnik, V., *Statistical Learning Theory*. John Wiley & Sons, 1998.

[106] Wang, H.-C., "MAT-a project to collect mandarin speech data through networks in Taiwan," *International Journal of Computational Linguistics and Chinese Language Processing*, pp. 73–89, February 1997.

[107] Wang, S., "Second-order necessary and sufficient conditions in multiobjective programming," *Numerical Functional Analysis and Optimization*, vol. 12, pp. 237–252, 1991.

[108] Xu, P. and Jelinek, F., "Random forests in language modeling," (Barcelona, Spain), pp. 325–332, 2004.

[109] Yaman, S., Chien, J.-T., and Lee, C.-H., "Structural Bayesian language modeling and adaptation," (Antwerp, Belgium), 2007.

[110] Yaman, S., Deng, L., Yu, D., Wang, Y.-Y., and Acero, A., "A discriminative training framework using n-best speech recognition transcriptions and scores for spoken utterance classification," in *the Proc. of International Conference on Acoustics, Speech, and Signal Processing*, vol. IV, (Honolulu, HI), pp. 5–8, April 2007.

[111] Yan, L., Dodier, R., Mozer, M. C., and Wolniewicz, R., "Optimizing classifier performance via an approximation to the Wilcoxon-Mann-Whitney statistic," in *in Proc. of International Conference on Machine Learning*, (Washington DC), 2003.

[112] Young, S. J., Jansen, J., Odell, J., Ollason, D., and Woodland, P. C., *The HTK book (for HTK version 2.0)*. Cambridge University, March 1996.

[113] Zissman, M. A., "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Transantions on Speech and Audio Processing*, vol. 4, no. 1, pp. 31–44, 1996.

# VITA

Sibel Yaman is a Ph.D. candidate in School of Electrical and Computer Engineering at Georgia Institute of Technology, Atlanta, GA, where she earned her M.S. degree in 2004. She received her B.S. degree in electrical and electronic engineering from Bilkent University, Ankara, Turkey in 2002. She is a recipient of the Microsoft Research Graduate Fellowship for 2006-2007. She has been selected as a Best Student Paper Award finalist in ICASSP 2006. Her research interests include automatic language identification, multi-objective optimization techniques for pattern classification, and language modeling.